$9.95

TEXAS INSTRUMENTS

BASIC PROGRAMMING FOR ADULTS

COMPUTER ADVANTAGE CLUB

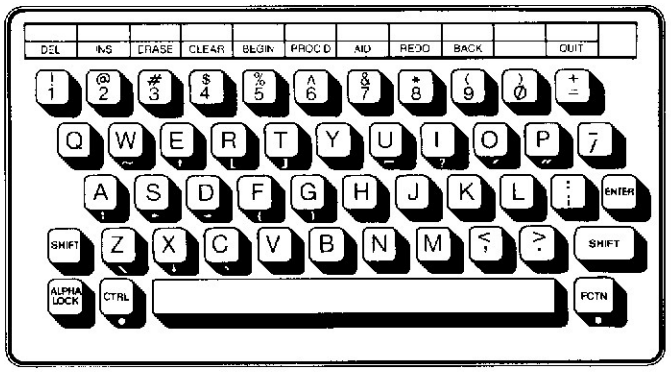
QUICK REFERENCE CHART OF TI-99/4A KEYBOARD FUNCTIONS

Press:

ALPHA LOCK
In locked (down) position causes characters on the screen to print in upper case; in unlocked (up) position produces lower case characters.

SPACE BAR
Moves cursor right, producing a space on the screen; also erases characters cursor moves across.

ENTER
Inputs information to the computer.



FCTN
The FCTN key accesses third characters on the front of the keys, as well as the functions identified on the interchangeable overlay strip located on the console above the keyboard. The FCTN key as well as the second key designated must be pressed simultaneously to access the desired function.

Press:

FCTN =	*(QUIT)*	Returns to TI Master Title Screen from program (or cartridge).
FCTN 1	*(DEL)*	Deletes text.
FCTN 2	*(INS)*	Prepares computer to insert text.
FCTN 3	*(ERASE)*	Clears an entire line from the screen.
FCTN 4	*(CLEAR)*	Causes a program in progress to stop.
FCTN S	*(◀)*	Moves cursor left without erasing text printed on the screen; negates DELete and INSert functions.
FCTN D	*(▶)*	Moves cursor right without erasing text printed on the screen; negates DELete and INSert functions.

IMMEDIATE MODE COMMANDS

The following commands are used frequently in Immediate Mode. CALL CLEAR can be used in both Immediate and Programming Modes.

Type:

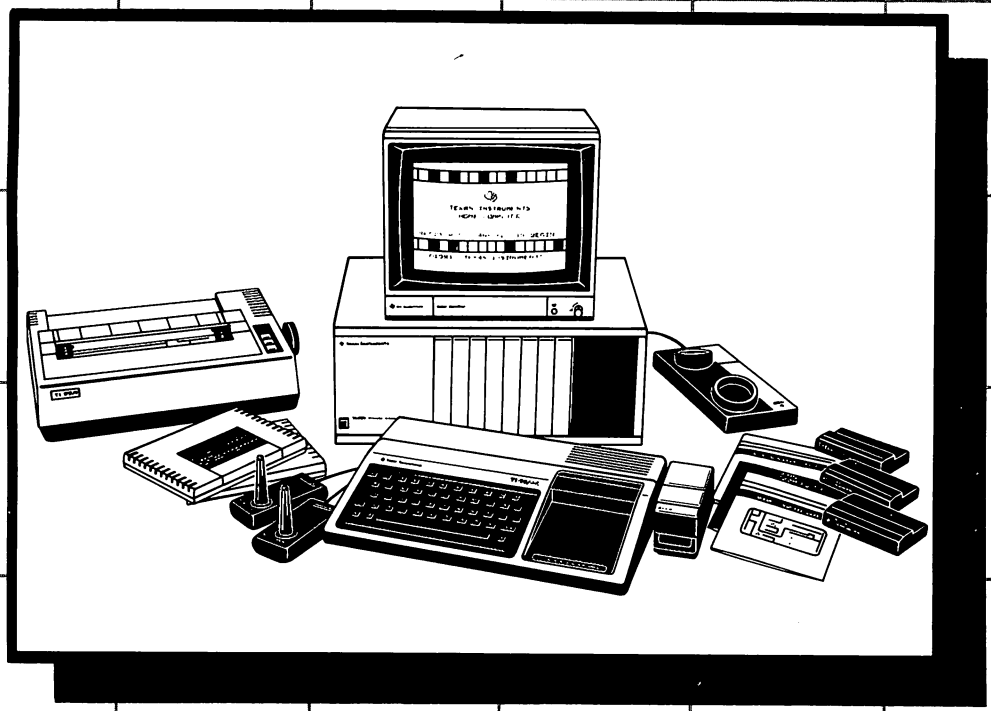
BYE	**Press:**	*(ENTER)*	Clears computer's memory and returns to TI Master Title Screen, leaving TI BASIC.
NEW		*(ENTER)*	Clears computer's memory and returns to TI BASIC READY screen (remains in TI BASIC).
CALL CLEAR		*(ENTER)*	Clears the screen but does not affect memory (remains in TI BASIC).
EDIT		*(ENTER)*	Brings the line, beginning with whatever number typed (100, 200, etc.) after pressing ENTER, to the screen for editing.
RUN		*(ENTER)*	Executes (runs) a program in memory.
LIST		*(ENTER)*	Lists a program in memory.

Developed by the staff of the
Texas Instruments Computer Advantage Club

TEXAS INSTRUMENTS

COMPUTER ADVANTAGE CLUB

BASIC PROGRAMMING FOR ADULTS



Texas Instruments invented the integrated circuit, the microprocessor, and the microcomputer, which have made TI synonymous with reliability, affordability, and compactness.

TABLE OF CONTENTS

PROGRAMMING LANGUAGES

Like any other language, a computer programming language is a means of exchanging information. Many different languages are used in computer programming. The following is a brief overview of some of those languages.

BASIC AND TI BASIC

BASIC stands for Beginners All-purpose Symbolic Instruction Code. Because BASIC contains terms such as PRINT, GOTO, RUN, and END, which have meanings similar to the English equivalent, it is the most popular programming language in use today. BASIC, like English, may be "spoken" in various dialects. The BASIC dialect built in to the TI-99/4A console is called TI BASIC. It has the ability to produce music, noise, color, animated graphics, and speech. In addition, TI BASIC gives the computer a full range of programming capabilities for most home and personal applications. Many personal computer owners find that BASIC in any dialect is more than satisfactory for their programming needs.

TI EXTENDED BASIC

When a programming language with more complex capability is desired, TI Extended BASIC is the first in a series of increasingly specialized languages available for use with the computer. It is often used to program business and professional software, and contains commands faster than TI BASIC and allows greater control of data input and screen formatting. When used in conjunction with other peripherals, TI Extended BASIC allows programs of almost unlimited length to be written.

ASSEMBLY

When a computer program is written in Assembly, instructions to the computer are automatically converted from the symbolic language code to the computer's own machine code. Commands issued in Assembly require minimal translation by the computer, and programs run much faster than when written in higher-level languages requiring extensive command interpretation by the computer. Because a great deal of programming knowledge is required to use Assembly, it is not intended for the beginning programmer.

PASCAL

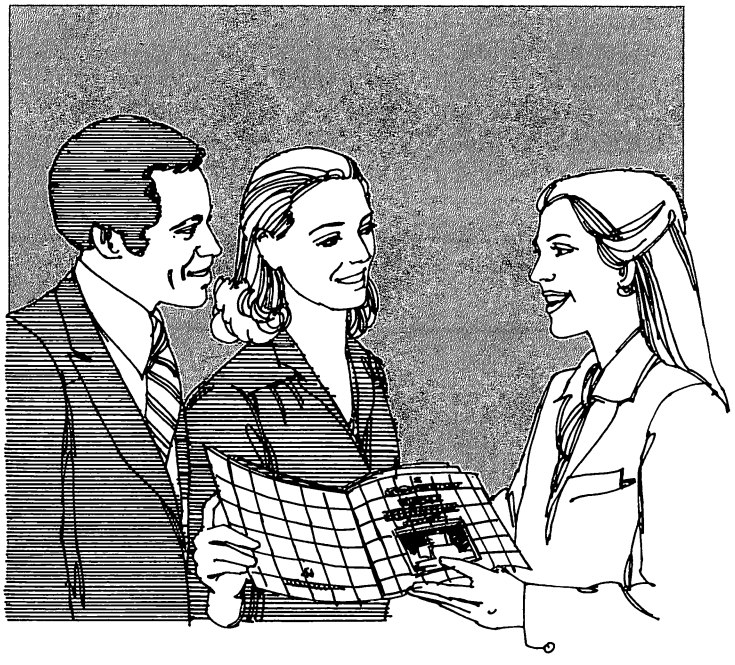
The Pascal programming language has the ability to access a large library of technical and professional programs, and has been specifically designed for transportability, i.e. to run on various types of computer systems. With the TI-99/4A system, instructions entered in Pascal are translated into "p" or "pseudo" code. Through use of the TI P-code Card peripheral, "p-code" is directly translated into the native machine code compatible with the computer.

TI LOGO

TI LOGO is easy to use and makes learning to program fun. The language is especially designed to create an open-ended learning environment for children who are just becoming acquainted with computers. Programming in TI LOGO allows you to "share" knowledge with the computer by giving it simple commands that return exciting graphics to the screen. TI LOGO's easy-to-understand commands encourage rapid progress from sketching, producing animated graphics, and writing, to mathematics and complex problem-solving. TI LOGO lets you experience the world of computer programming through self-paced exploration and discovery.

TI PILOT

PILOT is short for Programmed Inquiry, Learning, Or Teaching. Used to develop educational programs, PILOT programming language is employed specifically for Computer Assisted Instruction (CAI), and can be readily learned by instructors for expanded use in the classroom. With the growing use of computer-based instructional materials, educators are using computer programs as effective classroom tools. With TI PILOT, instructors can create programs to demonstrate concepts and simulate laboratory-like environments. Programming in TI PILOT makes features such as individualized drill, practice, and testing available.



GETTING STARTED: AN INTRODUCTION

The computer operates in one of two modes: Immediate Mode and Programming Mode. In ***Immediate Mode*** the computer carries out instructions or requested tasks the instant the ENTER key is pressed. In ***Programming Mode,*** the computer stores requested tasks in a series of numbered lines of instruction, called a ***program,*** until directed by the user to perform those tasks. When given the direction to proceed, the computer then executes the program by consecutive line number, or according to a specific branching instruction.

To review how computers work see page 40 in the appendix of this book. A tour of the keyboard and its functions is provided on the inside front cover. Computer components are outlined on the inside back cover. The computer ON/OFF switch is located on the front lower right corner of the console. When powering up the computer and its peripherals, the console should be turned on last. When powering down, the console should be turned off first.

The video screen is made up of 24 rows and 32 columns of lines (see the grids on p. 12). Each horizontal line on the screen can display up to 28 print positions and up to 32 graphics positions. Each printed statement may be four screen lines in length. When the cursor fills up one screen line, it automatically moves down to the next line, a process called ***wrapping,*** as you continue typing. The upward movement of printed lines on the screen is called ***scrolling.*** The TI-99/4A keyboard has a built-in ***automatic repeat*** feature. When the space bar or any character key is pressed down for longer than one second, the character prints repeatedly on the screen until the key is released.

There are a few situations in Immediate Mode which do not require use of the ENTER key. Two instances occur just after turning the system on. Turn on the TV or video monitor, then the computer console. The colorful TI Master Title Screen is displayed on the screen. As the message suggests, PRESS ANY KEY TO BEGIN. A selection list, or ***menu,*** appears. As indicated, PRESS 1 FOR TI BASIC. The message TI BASIC READY appears at the bottom of the screen. Below that message appears a prompt character (>) marking the beginning of each line typed, followed by a flashing cursor (■) indicating the computer is waiting for data, or ***input,*** to be entered.

ERROR MESSAGES

It is not uncommon to make errors when entering information to the computer. Whether it's a typographical error, an invalid or incomplete instruction (either in Immediate or Programming Mode), the computer will recognize the problem and print an ***error message*** on the screen. Some examples are INCORRECT STATEMENT, LINE TOO LONG, or CAN'T DO THAT. The comptuer will not proceed with the instruction until the error is corrected. Simply identify the error made, then retype the instruction correctly, or review the section Shortcuts and Editing Features (pp. 30-33) for further instructions on correction of errors. Refer to the *User's Reference Guide* pp. III-8-III-12 for an explanation of error messages.

KEYBOARD NOTES

The TI BASIC commands, statements, and functions introduced in this manual appear in capital letters (upper case). Although the computer will accept most TI BASIC commands, statements, and functions, in both upper and lower case, it is suggested that beginning programmers leave the ALPHA LOCK key in locked (down) position for upper case letters during programming sessions. Releasing the ALPHA LOCK key causes the TI-99/4A keyboard to function in a manner similar to a standard typewriter keyboard. It is important to remember that although the TI-99/4A keyboard is similar to a standard typewriter keyboard, **the computer only accepts a zero for numerical information and not the upper case letter O. Similarly, the numeral 1 and the lower case letter l cannot be used interchangeably.**

MATH EQUATIONS AND THE PRINT COMMAND

The computer performs simple calculator functions in Immediate Mode.

If you receive an error message after pressing ENTER, retype the line, correct any errors, and press ENTER again. The answer is displayed on the next line. Note how the screen printing scrolls upward when the answer is displayed.

Mathematical functions are performed using

- **+ for addition**
- **– for subtraction**
- *** for multiplication**
- **/ for division**
- **^ for exponentiation***

When executing mathematical operations, the computer performs exponentiation first, multiplication and division second, addition and subtraction third. Grouping with parentheses can change the order in which operations are performed, that is, operations enclosed within the innermost set of parentheses are performed first, those within the second innermost set are performed second, etc. These two examples illustrate how mathematical operations are affected.

```
PRINT 3+5
```
(Press ENTER)

The sum of the two digits "prints" on the screen.

```
PRINT 20+4/2+2^2
```
(Press ENTER)

The answer is 26 (20+2+4=26).

```
PRINT (20+4)/2+2^2
```
(Press ENTER)

The answer is 16 (24/2+4=16).

The PRINT command can cause not only numbers but also letters to print on the screen.

The computer will print any character, symbol, or space appearing between quotation marks when the PRINT command is used. Press FCTN P to produce quotation marks (located on the front of the P key).

```
PRINT "DON'T FORGET TO
PRESS ENTER!"
```
(Press ENTER)

The sentence without quotation marks prints on the screen.

The PRINT statement can position data printed on the screen with the use of colons, commas, and semicolons, called ***print separators.*** The examples illustrate the effect of each separator.

```
PRINT 1:2:3:4
```
(Press ENTER)

A colon is used to print data on separate lines. Each number prints on the next consecutive line.

```
PRINT 1,2,3,4
```
(Press ENTER)

A comma is used to separate data into two pre-positioned columns. The numbers 1 and 3 will print in the left-hand column (column position 2) on the screen, and 2 and 4

**Press the SHIFT key then the 6 key to produce the exponentiation symbol ^ (located on the number 6 key).*

will print in the right column (column position 16) on the screen. Because positive numbers are printed with a leading space, the numbers in this example print in columns 2 and 16.

PRINT "HELLO"; "HELLO"
(Press ENTER)

A semicolon causes adjacent print items to print side-by-side with no extra spaces between the values.

PRINT 1;2;3;4
(Press ENTER)

Although string variables (see pp. 6-7) are printed with neither leading nor trailing spaces, numbers are always printed with a trailing space. Positive numbers are printed with a leading space rather than with a plus sign. Negative numbers are printed with a leading minus sign in place of a leading space.

Positioning can also be performed using the TAB function.

The TAB function can be used to specify the column in which print items are to begin printing on the screen. The column is designated by the number within parentheses following the word TAB. TAB is very useful for printing two or more columns on the screen.

PRINT TAB(5);"HELLO"
(Press ENTER)

HELLO prints on the screen beginning at position 5.

VARIABLES

A *variable* is a letter, word, or group of letters and symbols representing memory locations in the computer. Variables can be assigned values through use of the LET statement. A ***numeric variable*** represents numbers. A ***string variable*** represents alphanumerics (letters, numbers, symbols, spaces).

Variable names may be up to fifteen characters in length, but must begin with a letter, an at sign (@), a left bracket ([), a right bracket (]), a back slash (/), or a line (_). A dollar sign ($) may appear only at the end of a string variable.

LET A=5
(Press ENTER)

A is a numeric variable assigned the numeric value of 5.

LET A$="HELLO"
(Press ENTER)

A$ is a string variable with the word HELLO as its assigned value. Note that string variables must end with a dollar sign, and the value of string variables must be enclosed in quotation marks.

The computer will accept variable assignments without the LET statement, therefore the use of LET is optional in TI BASIC.

Variables values can be retrieved or returned.

PRINT A
(Press ENTER)

The value of A (5) prints on the screen below the PRINT statement. If a value has not been entered for A, a zero will be printed.

PRINT A$
(Press ENTER)

The value of A$ (HELLO) prints on the screen below the PRINT statement. If a value has not been entered for A$, nothing will print on the screen.

To clear the screen of clutter or useless print, use the CALL CLEAR statement.

CALL CLEAR
(Press ENTER)

The screen clears, but you remain in TI BASIC.

PROGRAMMING MODE

***Line numbers* assigned to programming statements indicate a request to store the information for programming use.** When the ENTER key is pressed, that information is stored into the computer's memory. The computer waits until it receives the instruction to execute (RUN) the statements before acting upon them.

The program samples included in this book can help you explore TI BASIC. Study the program objectives and read the explanations before entering each line.

If you receive an error message after typing RUN and pressing ENTER, type LIST and press ENTER. The entire program will appear on the screen.

The following sections provide sample programs while introducing new programming statements.

Remember to follow these simple steps:

1. **Include a line number for each program statement.**
2. **Check each program line for typographical errors and correct them (see Shortcuts and Editing Features, pp. 30-33) before pressing ENTER.**
3. **After typing and correcting each line, press ENTER. This is the computer's cue to store the line in memory.**
4. **After typing and entering the entire program, type RUN, then press ENTER. The RUN command instructs the computer to perform or execute the program statements at that moment.**

Check each line carefully for:

- **Missing semicolons, quotation marks, etc.**
- **A misspelled statement such as PINT instead of PRINT**
- **Missing or incorrect line numbers, for example, 2 instead of 20, 2O instead of 20, l0 instead of 10**

Also, check to make sure the ALPHA LOCK key is in locked (down) position. See pp. 30-33 for further instructions on correcting errors and editing lines.

INPUT, GOTO, CALL SCREEN, CALL VCHAR, CALL HCHAR, FOR-NEXT, IF-THEN, READ-DATA, RND, RANDOMIZE, CALL SOUND, OPEN, CLOSE, CALL CHAR, GOSUB-RETURN, ON-GOSUB, ON-GOTO, LEN, SEG$, DIM, and CALL KEY. Read the statement explanations and type the programs. To the right of each new statement is a description of how the statement functions in the program.

Most important of all, have fun! Programming is an exciting new skill, and you may be surprised how quickly you get ***up and running*** *in TI BASIC.*

INPUT

The INPUT statement allows different values for one or more variables to be entered into a program. X and A$ are variables which will be assigned the value of the data entered. Remember to press ENTER after typing each program line. To execute the program in memory, remember to type RUN and press ENTER.

```
INPUT X
(for numeric values)
INPUT A$
(for string values)
10 PRINT "THANK YOU FOR
YOUR INPUT, BARBARA."
(Press ENTER)
RUN
(Press ENTER)
```

The sentence in quotation marks in line 10 prints on the screen.

The message between quotation marks is printed on the screen when the program is executed. To change the name in the message, either retype the entire line or edit it (see Shortcuts and Editing Features, pp. 30-33).

In the example, any name can be entered at line 20, which will then display on the screen accompanying the message in line 30. Using an INPUT statement causes the program to stop when it comes to the INPUT statement, display a question mark, and wait for input (in this case, a name) before continuing.

```
10 PRINT "PLEASE ENTER
YOUR FIRST NAME: "
(ENTER)
```

Line 10 prints instructions on the screen.

```
20 INPUT A$
(ENTER)
```

Line 20 displays a question mark and a flashing cursor, and waits for a name to be entered.

```
30 PRINT "THANK YOU FOR
YOUR INPUT, ";A$;"."
(ENTER)
```

Line 30 prints a message including the name entered in line 20.

```
RUN
(ENTER)
```

Executes the program.

If you receive an error message after typing RUN and pressing ENTER, type LIST and press ENTER. The entire program will appear on the screen. Check each line carefully for:

- **Missing semicolons, quotation marks, etc.**
- **A misspelled statement (PINT instead of PRINT)**
- **Missing or incorrect line numbers (2 instead of 20, 2O instead of 20, lO instead of 10)**

Also make sure the ALPHA LOCK key is in locked (down) position. See pp. 30-33 for further instructions on correcting errors and editing program lines.

An INPUT line can contain a message supplying a clue to the type of information expected.

An input prompting message rather than a question mark is displayed on the screen and the program waits for input before continuing. Line 30 is retained in the computer's memory to retain the final response. To view the program at this point, type LIST and press ENTER.

```
10 INPUT "PLEASE ENTER
YOUR FIRST NAME: ":A$
(ENTER)
```

Line 10 replaces the PRINT statement in line 10 of the previous example. Typing the new line 10 and pressing ENTER automatically replaces the former line 10. When this program is run, the instructions display on the screen and the flashing cursor prompts you to enter a name.

```
20
(ENTER)
```

Line 20, no longer required, is erased by typing the line number and pressing ENTER.

```
RUN
(ENTER)
```

Executes the program.

GOTO

The GOTO statement allows a program to *branch* to a line out of sequence. The computer begins at line 10, prints the message, then continues to line 20. There it is instructed to go back to line 10, and so on, creating a ***continuous loop*** causing the sentence on the screen to be printed repeatedly.

To stop the program, either turn the computer off, or press FCTN 4 (CLEAR) at which time the computer beeps. A ***breakpoint message*** displays on the screen (*BREAKPOINT AT 20) identifying the line at which the program stopped. To continue running the program from the point at which it stopped, type CONTINUE and press ENTER. To run the program again from the beginning, type RUN and press ENTER. For more information on CONTINUE and RUN commands, see pages 30-33.

NEW *(ENTER)*	Clears the computer's memory before beginning a new program.
10 PRINT "I AM THE TI-99/4A COMPUTER." *(ENTER)*	Line 10 prints a message.
20 GOTO 10 *(ENTER)*	Line 20 sends the program to line 10.
RUN *(ENTER)*	Executes the program.

CALL SCREEN

The CALL SCREEN statement can be used to change the color of the screen. There are sixteen different colors available in the system and each possesses a specific numeric code. To "call" a specific color to the screen, use the CALL SCREEN command followed by the number in parentheses corresponding to the color desired (see Color Code Chart, p. 34).

To change the color code number on line 10, first stop the program, breaking the loop, by pressing FCTN 4 (CLEAR). Type EDIT 10, press ENTER, and line 10 will appear on the screen. Use the right arrow key (FCTN D) to move the cursor to the color code number. Type the new color code number over the former number and press ENTER. Then type RUN to execute the program. Try different colors using the Color Code Chart for reference.

NEW	
10 CALL SCREEN(5) *(ENTER)*	Line 10 changes the screen color to dark blue, the color represented by the number 5.
20 GOTO 20	Line 20 sends the program repeatedly to line 20, creating a loop which holds the designated color on the screen indefinitely.
RUN	

This program changes the color of the screen depending on the number input while the program is running. If a number other than a number from 3 through 16 is entered, the program will not operate properly. If this happens, press FCTN 4 (CLEAR), press ENTER, and RUN the program again.

To stop the program, press FCTN 4 (CLEAR).

It is important to note that this program requires a ***numeric input*** for X in line 30. If something other than a number is entered, the program will stop and display an error message which reads WARNING: INPUT ERROR IN 30 TRY AGAIN: and will continue to do so until a number is entered.

```
NEW
10 CALL CLEAR
```
Line 10 clears the screen, but not the memory.

```
20 PRINT "ENTER A NUMBER FROM 3 TO 16, THEN PRESS ENTER:"
```
Line 20 prints directions on the screen to the user.

```
30 INPUT X
```
Line 30 allows the variable X, in this case a number, to be entered into the program. A question mark is displayed and the program waits for a number to be entered.

```
40 CALL SCREEN(X)
```
Line 40 changes the screen color according to the number entered in line 30.

```
50 GOTO 20
```
Line 50 loops the program back to line 20.

```
RUN
```

CALL VCHAR AND HCHAR

To create pictures or graphics on the screen in TI BASIC, the VCHAR (for "vertical character") and/or HCHAR (for "horizontal character") statements must be used. The screen can be viewed as a ***grid*** of square blocks consisting of 32 columns and 24 rows. Each character appearing on the screen exists in its own block or box consisting of an 8-by-8 matrix (see the grid on p. 21).

The VCHAR and HCHAR statements must include numbers separated by commas, enclosed in parentheses. The program statement in line 10 calls a specific character to a designated point on the screen.

The first number in the set (12) is the row number, the second (17) identifies the column number, and the third (42) represents the character code number.

The VCHAR statement in line 20 produces a vertical column of asterisks.

```
NEW
10 CALL VCHAR(12,15,42)
```
Line 10 causes an asterisk (character code 42, see chart p. 34) to appear on the screen in row 12, column 15.

```
20 CALL VCHAR(1,15,42,24)
```
Line 20 causes 24 asterisks to appear on the screen beginning at row 1 column 15. The fourth number in the set specifies the number of times a character is to be repeated.

The HCHAR statement in line 30 works in a similar manner producing horizontal rows.

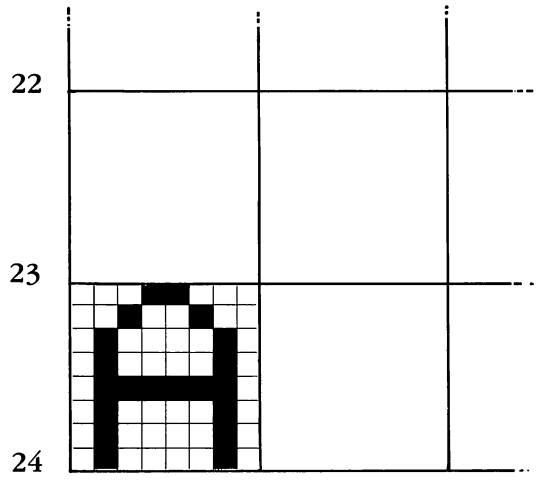
```
30 CALL HCHAR(12,1,42,32)
```

Line 30 produces a row of 32 asterisks across the screen beginning at row 12 column 1.

```
RUN
```

Refer to the Character Set Codes on page 34 to produce your own graphics. Two blank grids are provided on the following page for sketching designs.

Each print character is composed of dots in an 8-by-8 matrix, that is, 8 rows of 8 dots each. The character you actually see, such as an asterisk, can take on a specific color, called the ***foreground color.*** The 8-by-8 character block can also take on color, called the ***background color.***




```
NEW
10 CALL CLEAR
20 CALL VCHAR(1,15,42,24)
30 CALL HCHAR(12,1,42,32)
40 CALL COLOR(2,7,12)
50 GOTO 50
RUN
```

Line 10 replaces the former line 10.

Line 40 assigns character set 2 with a foreground color 7 (dark red) and a background color 12 (light yellow).

Line 50 produces a loop causing the program to remain on the screen. Without line 50, the colors indicated in the CALL COLOR statement flash only momentarily on the screen.

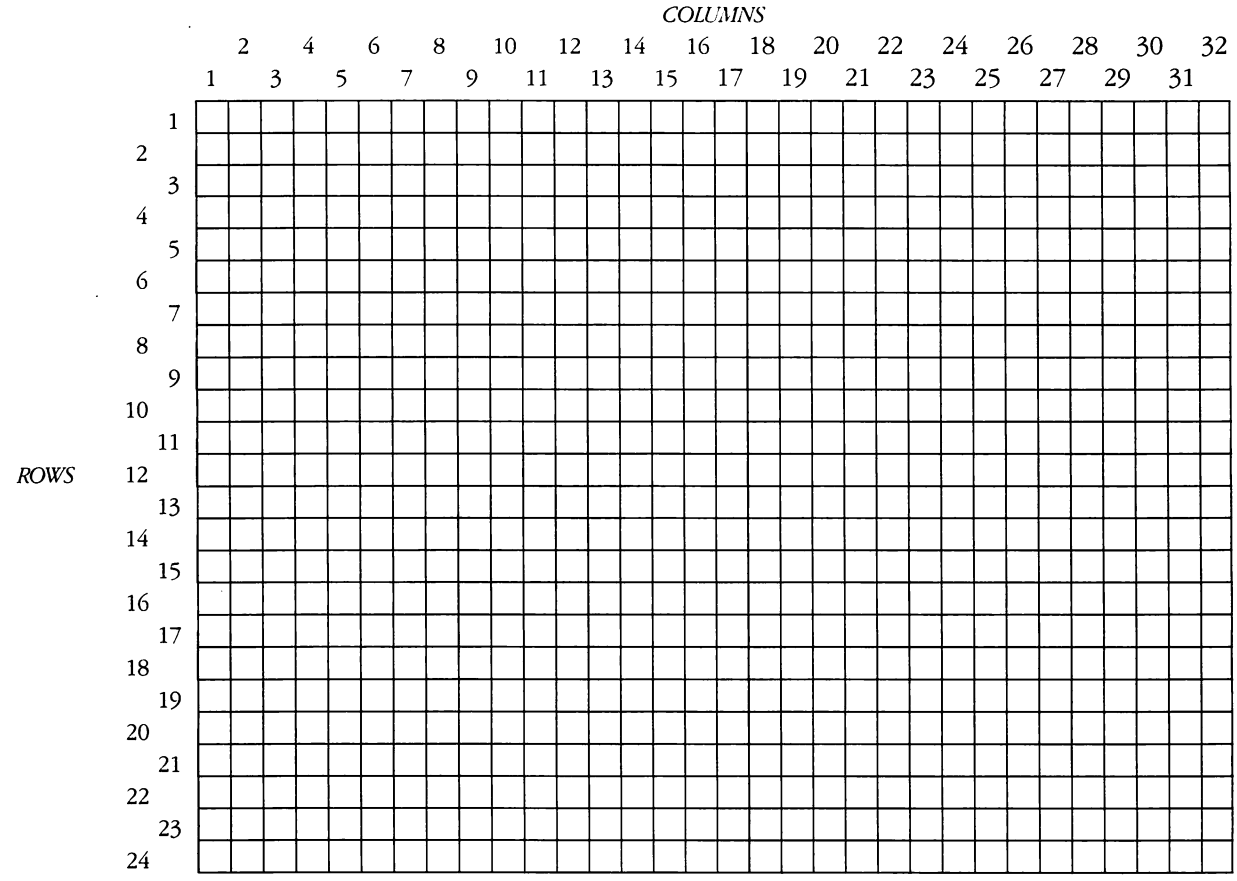
The VCHAR or HCHAR, as well as the background color in the VCHAR or HCHAR block, can be varied using one program statement—the CALL COLOR statement. This statement requires three numbers separated by commas, enclosed in parentheses, as in line 40.

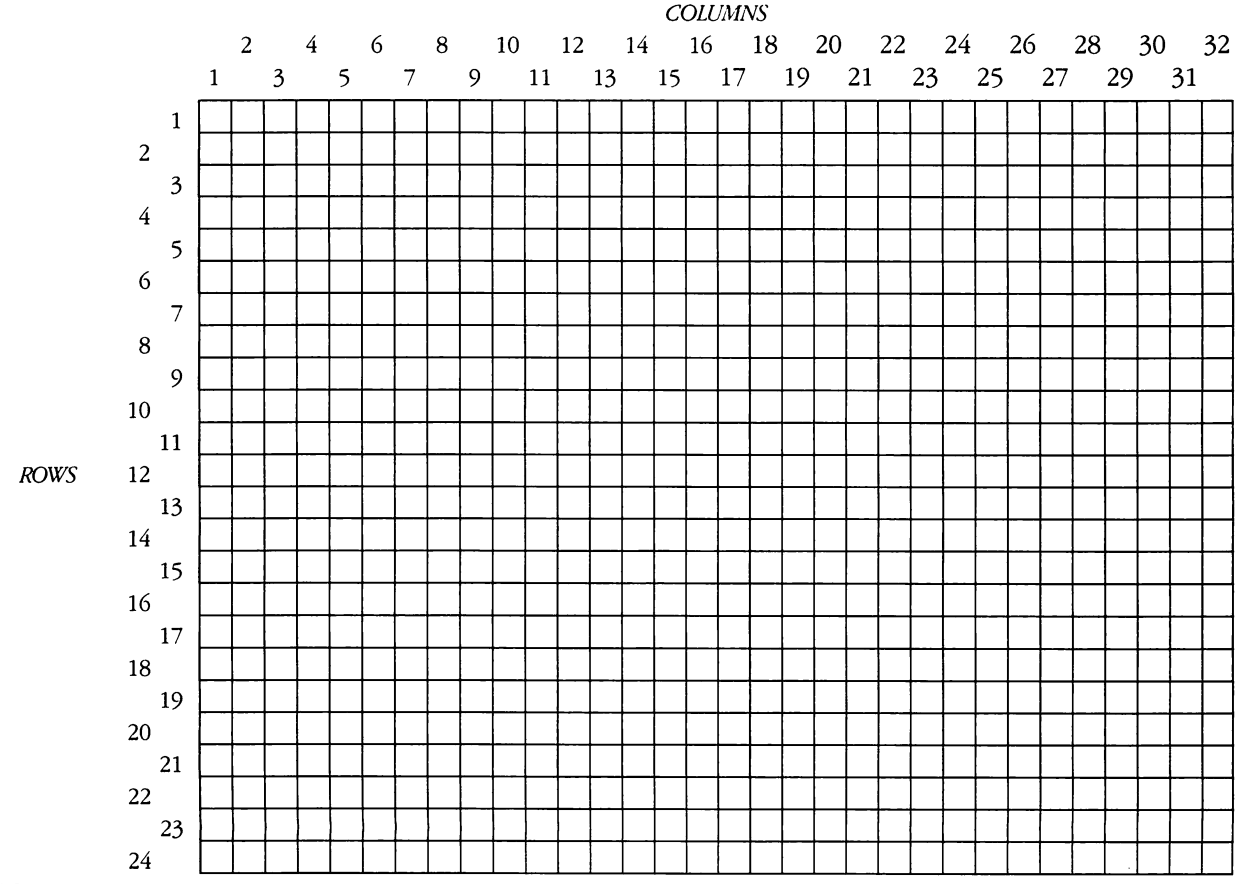
The first number (2) is the ***character set number*** (see chart p. 34). The asterisk, or character number 42, is included in character set #2. The second number (7) sets the foreground or character color (dark red). The third number (12) sets the background color (light yellow) in which the character appears.

To change this program in any way, press FCTN 4 (CLEAR) to stop the program, then begin editing (see pp. 30-33 for editing instructions).

Using the Color Code and Character Code Charts (p. 34) for reference, designs of your choice can be created with combinations of HCHAR, VCHAR, CALL COLOR, and CALL SCREEN.

GRAPHING WORKSHEETS





FOR-NEXT

In preceding sections the GOTO statement was introduced and briefly explained. GOTO creates a continuous loop within a program by unconditionally returning to the line specified in the GOTO statement. This is also called ***unconditional branching.*** **The FOR-NEXT statement creates a controlled loop where the program loops the number of times specified in the FOR statement, and loops only between the FOR and NEXT statements.** This action is referred to as ***conditional branching.*** For every FOR statement in a program, there must also be a NEXT statement.

```
NEW
10 FOR X=1 TO 25
```

Line 10 sets up a loop to repeat 25 times.

```
20 PRINT "THIS PROGRAM WILL RUN";(25-X); "MORE TIMES"
```

Line 20 prints on the screen the number of times the program has left to repeat.

```
30 NEXT X
```

Line 30 sends the program back to line 10 for the next repeat.

```
40 END
RUN
```

Line 40 terminates the program.

```
NEW
10 CALL CLEAR
20 CALL COLOR(2,7,7)
30 CALL HCHAR(12,3,42,28)
40 FOR B=1 TO 1000
50 NEXT B
60 END
RUN
```

In lines 40 and 50, the FOR and NEXT statements set up a loop requiring the computer to count to 1000 before ending the program.

Can you predict the activity in this program?

```
NEW
10 CALL VCHAR(12,16,65)
20 FOR DELAY=1 TO 200
30 NEXT DELAY
```

Line 20 begins a pause by creating a loop requiring the computer to count to 200 before continuing the program. It does this by incrementing the value of DELAY by 1, and testing for the maximum assigned value (200). If the maximum assigned value has not been reached, the program continues to line 30, which sends the program back to line 20, and the process is repeated until the maximum assigned value is reached. When it is reached, the program continues on to line 40.

```
40 CALL CLEAR
50 FOR DELAY=1 TO 100
```

Line 50 sets up another loop creating a second pause.

```
60 NEXT DELAY
```

Line 60 sends the computer to line 50 continuing the second pause.

```
70 GOTO 10
```

Line 70 returns the program to its starting point to begin again.

```
RUN
```

FOR-NEXT loops may be ***nested,*** that is, one may be contained within another.

When using nested loops, observe the following rules:

1) Each FOR statement must be paired with a NEXT statement.
2) Each nested loop must use different control variables.
3) The inside loop (Y) must be contained within the outside loop (X).

```
NEW
10 FOR X=1 TO 3
20 FOR Y=1 TO 3
30 PRINT X;Y
40 NEXT Y
50 NEXT X
60 END
RUN
```

Line 10 sets up the first loop paired with line 50.

Line 20 sets up the second loop paired with line 40.

Line 30 prints the values of X and Y.

IF-THEN

The IF-THEN statement is used when a decision needs to be made within a program.

A "test" is set up within the IF-THEN statement. The results of the test determine the next line to be executed, often requiring the program to branch to another part of the program.

```
NEW
10 CALL CLEAR
20 LET K=1
30 PRINT "K=";K
40 LET K=K+1
50 IF K<10 THEN 30
60 PRINT "OUT OF LOOP"
70 END
RUN
```

Line 30 prints the value of K.

Line 40 increases the value of K by 1.

Line 50 sets up the test to determine if the value of K is less than 10. If so, the program branches to line 30. If not, the program continues to line 60.

Line 60 prints a message to the programmer that K's value has exceeded the designated maximum.

This program adds the ELSE condition to an IF-THEN statement. Lines 20 and 30 test the value of N and set up conditions. This program can determine whether or not the number entered is a number from 1 to 10. To test this program, try entering negative numbers, decimal fractions, and numbers greater than ten.

```
NEW
10 INPUT "ENTER A
NUMBER FROM 1 TO 10: ":N
20 IF N>10 THEN 40
30 IF N<1 THEN 40 ELSE 60
40 PRINT "BAD NUMBER.
TRY AGAIN."
50 GOTO 10
60 PRINT "VERY GOOD!"
70 END
RUN
```

Line 10 provides a message elaborating on the type of input needed for N.

Line 20 performs a test. If N is greater than 10, the program branches to line 40. If not, it proceeds to line 30 for the next test.

Line 30 performs another test. If N is less than 1, the program branches to line 40. If not, it proceeds to line 60.

Line 40 indicates a different input is necessary.

Line 60 indicates the correct choice was made.

READ-DATA

Some programs contain data which need to be stored within a program for later use. **The DATA statement is useful for storage of large amounts of information. The READ statement allows selective access to information stored in DATA statements.** This short program illustrates the use of READ-DATA statements.

```
NEW
10 READ A$,B$,C$
20 DATA JOE,MARY,BILL
30 PRINT A$:B$:C$
RUN
```

Line 10 assigns variable names to the information in the DATA statement (A$ = JOE, B$ = MARY, C$ = BILL).

Line 20 names JOE, MARY, and BILL as the data to be accessed.

Line 30 prints the data in the format desired (one name per line).

This program illustrates the same operation using standard LET statements to assign values to string variables.

```
NEW
10 LET A$="JOE"
20 LET B$="MARY"
30 LET C$="BILL"
40 PRINT A$
50 PRINT B$
60 PRINT C$
70 END
RUN
```

Lines 10-30 assign values to string variables.

Lines 40-60 print the values of the string variables on the screen.

READ-DATA statements can also be used to assign numeric variables to numeric values or data. Note that commas must separate data items assigned to numeric variables.

```
NEW
10 READ A,B,C
20 DATA 10.50,12.75,13.05
30 PRINT A:B:C
40 END
RUN
```

Line 10 identifies the numeric variables to be "read."

Line 20 identifies the values assigned to each variable respectively.

String variables can contain numeric data.

```
NEW
10 READ A$,B$,C$
20 DATA JOE 10.50, MARY
12.75, BILL 13.05
30 PRINT A$:B$:C$
40 END
RUN
```

Line 10 identifies the string variable to be "read."

Line 20 identifies the values assigned to each variable respectively.

A single string variable name may be assigned to an entire line of data. Note that commas are not used to separate data items assigned to string variables.

```
NEW
10 READ A$
20 DATA JOE 10.50 MARY 12.75 BILL 13.05
30 PRINT A$
40 END
RUN
```

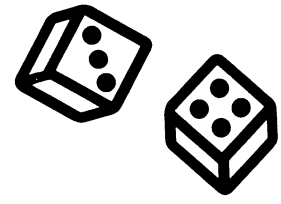
Both numeric and string data can be contained in string variables, but only numeric data may be contained in numeric variables. Also, numeric data contained in a string variable cannot be accessed by the computer for mathematical operations.

READ-DATA can also be used to assign values to a variable in a sequential manner. For more information on the READ-DATA statements see the *User's Reference Guide*, pp. II-61–II-63.

RANDOM NUMBERS (RND) AND RANDOMIZE

The RND statement will generate a random number. This function is useful in the creation of programs for games, simulations (such as the roll of dice), graphics, and music programs. To try RND, ENTER the line PRINT RND in Immediate Mode (no line numbers necessary) several times. Each time RND is entered, a random number is generated and displayed. Unless a range of numbers is designated, the RND statement will always generate a number greater than or equal to zero and less than one. A range can be assigned to a random number set to generate random numbers other than those between 0 and 1.

If only whole numbers are desired, use the INTeger statement in conjunction with RND. To obtain a range from 1 to 10, the value of 1 must be added to the statement.



```
NEW
10 PRINT INT(RND*10)
20 GOTO 10
RUN
```

Generates integers from 0 to 9.

```
NEW
10 PRINT INT(RND*10)+ 1
20 GOTO 10
RUN
```

Generates integers in a range from 1 to 10.

This "guessing game" program illustrates these principles. Run this program several times. To stop program execution, press FCTN 4 (CLEAR). Note that the sequence of numbers is the same.

```
NEW
10 X=INT(RND*10)+1
20 INPUT "ENTER A NUMBER BETWEEN 1 AND 10: ":N
30 IF N>X THEN 70
40 IF N<X THEN 90
50 PRINT "YOU GOT IT!"
60 GOTO 10
70 PRINT "YOUR GUESS IS TOO HIGH"
80 GOTO 20
90 PRINT "YOUR GUESS IS TOO LOW"
100 GOTO 20
110 END
RUN
```

Line 10 tells the computer to generate a random number between 1 and 10.

Line 20 prints a question mark and waits for a number to be entered.

Lines 30-40 test the "guess" and branch to other parts of the program if the guess is not correct.

Lines 50-100 display the results of the test in lines 30-40 and create loops in the program.

To create random sequencing, add the RANDOMIZE statement in another program line preceding the INTeger statement.

```
5 RANDOMIZE
```

Line 5 creates random sequencing. This statement must precede the INTeger statement.

CALL SOUND

The TI-99/4A has built-in capabilities to produce five octaves of musical tones–either individually, in chords, or in tunes–and non-musical sounds (noise) using the CALL SOUND statement. The CALL SOUND statement must be followed by a set of specific numbers separated by commas and enclosed in parentheses.

Before running this sample program, be sure the volume is audible. Also make sure the ALPHA LOCK key is in locked (down) position.

The first number (1000) sets the duration, which can range from 1 to 4250 milliseconds, inclusive. The second number (440) sets the

```
NEW
10 CALL SOUND (1000,440,2)
```

Line 10 produces a single musical tone.

tone frequency, which can range from 110 to 44733, inclusive. The third number (2) sets the volume, which can range from 0 (loudest) to 30 (quietest), inclusive. This program line produces the single musical tone A above middle C. See the Musical Tones Frequencies chart on page 34 for further experimentation.

Up to three tones can be produced per statement.

```
20 CALL SOUND
(1000,440,2,659,2,880,2)
```

Line 20 produces a musical chord.

The duration can apply to up to three tones. All tones must carry the same duration per CALL SOUND statement. The first tone is set by 440,2, the second by 659,2, and the third by 880,2.

Noise can be generated by inserting a negative integer from −1 to −8 in the second position.

```
30 CALL SOUND(1000, -2,2)
RUN
```

Line 30 produces a noise.

TI's *Beginner's BASIC* manual is a good reference for programming music and noise.

METRONOME, ANYONE?

This program causes the computer to simulate a metronome, a device used to mark time at a steady beat in adjustable intervals.

To stop program execution, press FCTN 4 (CLEAR). The tempo of the beat can be varied by editing line 110 and giving DELAY a different value. A value of 1 to 5, for example, produces a very fast or prestissimo tempo. A value of 1 to 200 produces a slow or largo tempo.

The sound of the "click" can be changed by editing line 100 and varying the duration, tone, and volume values. Some combinations can even produce sounds similar to percussion instruments.

Now your computer's "got the beat!"

```
NEW
100 CALL SOUND(8, -5,20)
110 FOR DELAY=1 TO 90
120 NEXT DELAY
130 GOTO 100
RUN
```

SOUND EFFECTS

Run these programs to experience realistic sound effects. Try to understand the purpose of each program statement and its relation to the other statements.

***Going Up**

```
NEW
100 FOR X=110 TO 10000 STEP 25
110 CALL SOUND(-100,X,0)
120 NEXT X
RUN
```

Crash

```
NEW
100 CALL SOUND
(2000,-7,0)
RUN
```

***Hills and Valleys**

```
NEW
10 FOR X=110 TO 2000 STEP 25
20 CALL SOUND(-100,X,0)
30 NEXT X
40 FOR X=2000 TO 110 STEP -25
50 CALL SOUND(-100,X,0)
60 NEXT X
70 GOTO 10
RUN
```

*See page 26 for explanation of the STEP statement

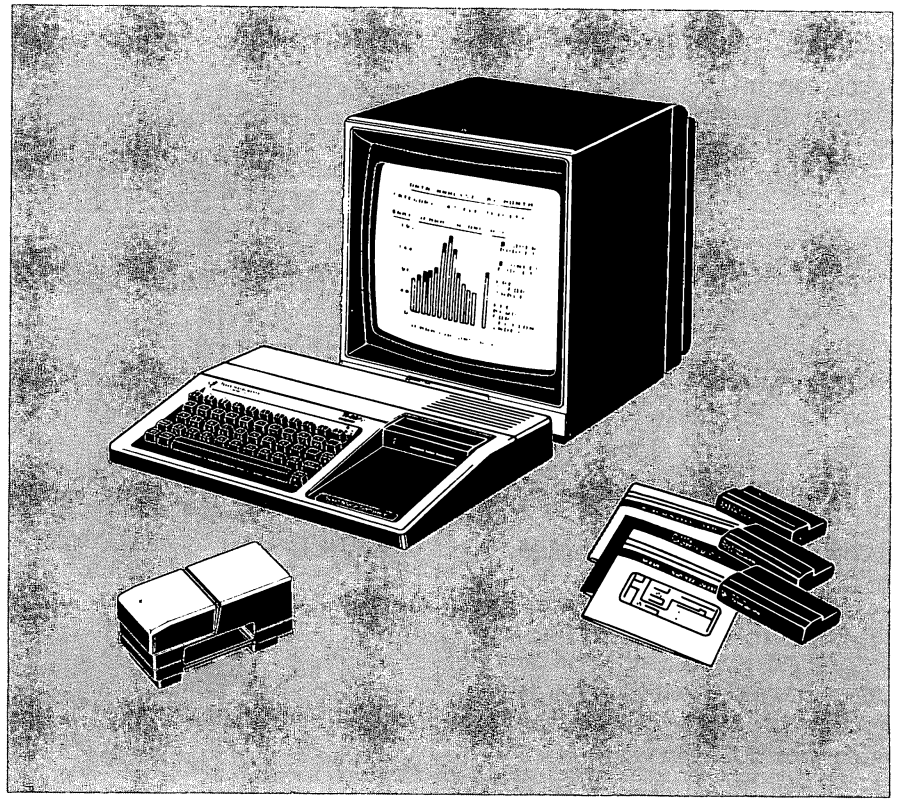
SPEECH

The TI-99/4A computer has speech capabilities. It is not prerecorded speech, but speech generated from combining strings of minimal speech units called "allophones." Type whatever you want the computer to say, and the computer will say it! This function is called text-to-speech.

To access text-to-speech capabilities, the following system is required:

- A TI-99/4A console
- A color TV or monitor
- A Speech Synthesizer unit*
- A Terminal Emulator II cartridge*

To begin a programming session after the Terminal Emulator II cartridge has been inserted, press any key to begin, then press 1 for TI BASIC. Check to make sure the ALPHA LOCK key is in locked (down) position before entering program lines. The computer will display an error message when the program is run if speech statements in programs are entered in lower case rather than upper case letters.



*See guide accompanying units for assembly or insertion instructions.

To incorporate speech in a BASIC program, an OPEN and a PRINT statement must be used. It is good practice to include a CLOSE statement in a program whenever an OPEN statement is used.

```
NEW
100 OPEN #1:"SPEECH", OUTPUT
```

Line 100 enables the computer to access the information contained in the Speech Synthesizer unit.

```
110 PRINT #1:"I CAN NOW SPEAK UP TO FOUR LINES OF WHATEVER YOU CHOOSE. A RATHER IMPRESSIVE FEATURE, ISN'T IT?"
```

Line 110 causes the message between quotation marks to be output as speech rather than text printed on the screen.

```
300 CLOSE #1
```

Line 300 closes the OPEN file in line 100.

```
310 END
```

Line 310 teminates the program.

```
RUN
```

One statement line is limited to about four lines of a message, as in line 110. To include more lines of speech in a program, simply add more program lines with the statement PRINT #1.

The misspellings which occur in the text of the program above are not accidental. As you work more with synthetic speech, you will find that an occasional incorrect spelling will produce clearer speech.

```
NEW
100 OPEN #1:"SPEECH", OUTPUT
110 PRINT #1:"HELLO. I AM THE TEXIS INSTRUHMENTS TALKING HOME COMPUTER."
120 PRINT #1:"WHERE DO I COME FROM, YOU ASK? WHY, I WAS BUILT IN LUBBUHCK, TEXIS."
130 PRINT #1:"WHY DON'T I HAVE A TEXIS ACCENT, YOU SAY?"
140 PRINT #1:"BECAUSE THE PEOPLE WHO PROGRAMMED MY SPEECH TALK JUST LIKE ME."
300 CLOSE #1
310 END
RUN
```

To practice with the TI-99/4A speech capabilities, type and run the following program.

```
NEW
100 OPEN #1:
"SPEECH",OUTPUT
```

Line 100 tells the computer to access the speech unit.

```
110 INPUT "TYPE WHAT YOU WANT ME TO SAY, THEN PRESS ENTER: ":A$
```

Line 110 allows you to type in what you want the computer to say.

```
120 PRINT #1:A$
```

Line 120 tells the computer to say whatever you typed.

```
130 GOTO 110
```

Line 130 tells the computer to repeat the process by saying whatever you type in next.

```
RUN
```

To change the computer's voice, use the formula //xx yyy.

```
110 //20 100
```

Replaces the former line 110 that requested input.

The next phrase input for the computer to speak will be in a higher-pitched voice than the normal computer voice. To stop program execution, press FCTN 4 (CLEAR). If the program is stopped and run again, the voice returns to its normal values.

Experiment with other voices. In the format //xx yyy, xx sets the pitch of the voice and yyy sets the slope (rising or falling intonation). For xx, use a number between 0 and 63. The lowest-pitched voice is 63, the highest is 1, and a whisper is 0. For yyy, use a value from 0 through 255. The standard pitch and slope for the TI-99/4A voice is 43 and 128, respectively. The best results occur when the slope is 32 times 10% of the pitch.

The pitch and stress of the speech produced can also be altered. For more detailed information regarding the flexibility of TI synthetic speech, refer to the Terminal Emulator II manual.

More than one voice can be produced in a program incorporating speech. Experiment with different voice qualities using the //xx yyy notation. Include the notation for each voice on a separate line preceding the line containing the text each voice is to speak.

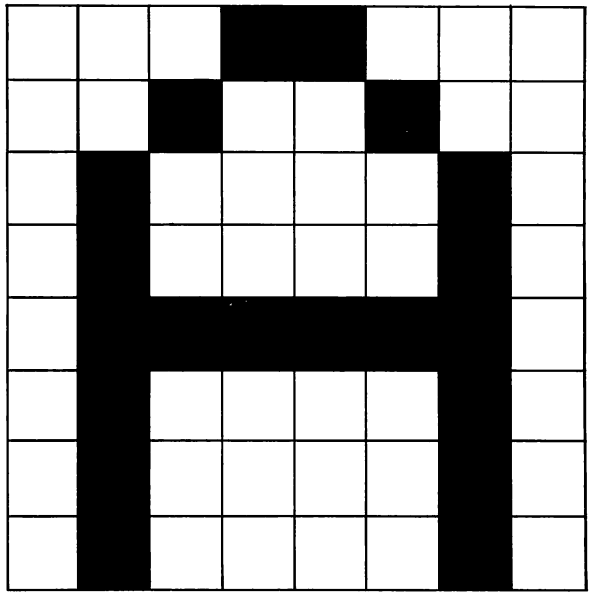
The ability to incorporate speech in programs provides not only state-of-the-art technology, but also many hours of programming fun!

```
NEW
100 OPEN #1: "SPEECH", OUTPUT
110 CALL CLEAR
120 PRINT TAB(10); "RING . . . RING"::::::::::
130 CALL CLEAR
140 PRINT #1: "//00 100"
150 PRINT #1: "HELLO. MAY I HELP YOU?"
160 PRINT #1: "//20 100"
170 PRINT #1: "UH. NO. I THINK I HAVE THE WRONG
NUMBER."
300 CLOSE #1
RUN
```

CALL CHAR

The CALL CHAR statement allows non-standard screen characters to be created. By using this statement, standard keyboard characters are redefined. Each character or printing position on the screen can be considered a square or grid of usable space composed of 64 dots. The dots are arranged in an 8-by-8 grid (8 rows of 8 dots each) as illustrated for the letter A.

A character on the screen is formed by turning specific dots within the matrix "on" and leaving the others "off." The actual process by which the dots used in a given matrix are selected and coded is a very involved process. An in-depth discussion of CALL CHAR can be found in *Beginner's BASIC* (pages 108-117) and the *User's Reference Guide* (pages II-76–II-79).



This program provides an illustration of the CALL CHAR statement where the standard character 96 (a grave accent) is redefined as a character that looks like a square within a square.

```
NEW
10 CALL CLEAR
20 CALL VCHAR(3,16,96)
30 CALL CHAR (96,
"FFFFC3C3C3FFFF")

40 GOTO 40
RUN
```

Line 30 redefines the shape of character 96 (a grave accent) to the "square within a square" character defined by FFFFC3C3C3FFFF. When this program runs, a small square appears in the upper center of the screen.

GOSUB-RETURN (Subroutines)

The GOSUB statement allows a program to branch to a designated section within the program. Upon encountering a RETURN statement, the program goes back to the line following the GOSUB statement and proceeds from there. This program is a simple but effective illustration of how GOSUB and RETURN operate within a program.

```
NEW
100 GOSUB 400
```

Line 100 sends the program to a subroutine in line 400, that clears the screen. When the RETURN statement is encountered on line 410,the program returns to the line following the GOSUB statement (line110).

```
110 PRINT "TO BEGIN A
PROGRAMMING SESSION IN
TI BASIC, FOLLOW THIS
SIMPLE PROCEDURE:"
```

Line 110 prints a message.

```
120 GOSUB 300
```

Line 120 sends the program to the subroutine on line 300, a FOR-NEXT loop that causes a pause holding the message in line 100 on the screen for a few seconds. The RETURN in line 320 sends the program to line 130.

```
130 GOSUB 400
```

Line 130 sends the program to the CALL CLEAR subroutine at line 400 again. Line 410 returns the program to line 140.

```
140 PRINT "1) TURN ON ALL
PERIPHERALS.":::::
```

Line 140 prints a message.

```
150 GOSUB 300
160 GOSUB 400
170 PRINT "2) TURN ON THE
CONSOLE.":::::
180 GOSUB 300
190 GOSUB 400
200 PRINT "3) PRESS ANY
KEY TO BEGIN.":::::
210 GOSUB 300
220 GOSUB 400
230 PRINT "4) PRESS 1 FOR TI BASIC.":::::
240 GOSUB 300
250 GOSUB 400
260 END
300 FOR DELAY=1 TO 700
310 NEXT DELAY
320 RETURN
400 CALL CLEAR
410 RETURN
RUN
```

Line 150 sends the program again to the subroutine in line 300, and the program continues executing in this order: lines 300, 310, 160, 400, 410, 170, 180, 300, 310, 190, 400, 410, 200, 210, 300, 310, 220, 400, 410, 230, 240, 300, 310, 250, 400, 410, and ends with line 260.

Subroutines are useful for saving space in a program by compacting otherwise repetitious routines into one section of the program.

ON-GOSUB

The ON-GOSUB statement allows a program to branch to one of several subroutines in a program. The format of the ON-GOSUB statement is ON X GOSUB Y,Z. X is a numeric expression and Y and Z are program line numbers.

In this tax program, subroutines are found on lines 300, 400, and 500. The appropriate subroutine is selected in the program when an input of 1, 2, or 3 is entered in line 180. An input of 1 is assigned the first line number in the ON-GOSUB statement (line 200), 2 is assigned the second line number, and so on. The program then branches to the line number corresponding to the value input, and following the ON-GOSUB statement.

```
NEW
100 CALL CLEAR
110 INPUT "ENTER
PURCHASE PRICE: $":PRICE
120 PRINT :::
130 PRINT "SALES TAX
CODES"
140 PRINT
"_____"
150 PRINT "ENTER 1 FOR
4% TAX"
160 PRINT "ENTER 2 FOR
5% TAX"
170 PRINT "ENTER 3 FOR
6% TAX"
180 INPUT "TAX SELEC
TION? ":TAX
190 IF TAX > 3 THEN 180
ELSE 200
200 ON TAX GOSUB
300,400,500
210 TAX = INT
(PRICE*RATE*100+.5)/100
220 TOTAL=TAX+PRICE
230 GOTO 600
290 REM TAX CODE
SUBROUTINES
300 RATE=.04
310 RETURN
400 RATE=.05
```

Line 110 asks for a dollar amount to be input.

Lines 130-170 set up a screen detailing sales tax codes.

Line 180 asks for a tax code input.

Line 190 is a testing protection device sending the program back to input line 180 if a number larger than 3 is entered.

Line 200 sends the program to a subroutine on line 300 if 1 is input, to line 400 if 2 is input, to line 500 if 3 is input. Upon encountering a RETURN (lines 310, 410, or 510) the program returns to line 210, where tax is computed according to the rate selected.

Line 220 computes the total cost of purchase.

Line 230 sends the program to the output format section (lines 600-680), where the initial purchase amount, tax, and total of tax and purchase amount are displayed on the screen.

The REMark statement allows notes to be supplied within a program when it is listed. REMark notes will not display on the screen when the program is run.

```
410 RETURN
500 RATE=.06
510 RETURN
590 REM OUTPUT FORMAT
600 CALL CLEAR
610 PRINT "PURCHASE PRICE";TAB(17);PRICE
620 PRINT
630 PRINT "PLUS";RATE*100;"% TAX";TAB(18);TAX
640 PRINT
650 PRINT TAB(17);"_________"
660 PRINT ::
670 PRINT "TOTAL";TAB(17);TOTAL
680 PRINT :::::
690 PRINT "PRESS 1 FOR
ANOTHER INPUT"
700 PRINT "PRESS 2 TO
QUIT"
710 INPUT "CHOICE? ":X
720 IF X > 2 THEN 710
730 ON X GOTO 100,800
800 END
RUN
```

Lines 690-710 ask if the user wishes to perform another computation or to quit.

Line 720 protects against input greater than 2.

Line 730 sends the program to line 100 if 1 is pressed, or to line 800 if 2 is pressed.

ON-GOSUB is useful for branching after a displayed menu or selection list.

ON-GOTO

The ON-GOTO statement instructs the computer to branch to one of several program lines, depending on the value of the numeric expression which must be placed between ON and GOTO.

In the example, if 1, 2, or 3 is input in line 100, the program branches to lines 120, 140, or 160, respectively. If a number larger than 3 is entered, an error message (BAD LINE NUMBER) is displayed.

```
NEW
100 INPUT "ENTER 1, 2 OR
3 ":A
110 ON A GOTO 120, 140,
160
120 PRINT "A=1"
130 GOTO 100
140 PRINT "A=2"
150 GOTO 100
160 PRINT "A=3"
170 GOTO 100
RUN
```

To stop program execution press FCTN 4 (CLEAR). ON-GOTO is very useful for branching after a displayed menu or selection list.

LEN

The LEN (for "length") statement counts the number of characters in a specific string expression and displays the number on the screen.

To be counted, a string variable must be enclosed within parentheses and follow the LEN statement.

When this program is run, the string expression (the value of A$) will be printed on the screen, as directed by line 110. Then the number of characters in the string expression (40) is printed on the screen as directed by line 120. All spaces and punctuation are counted as characters in the string expression.

```
NEW
100 A$="HELLO THERE, I AM THE TI-99/4A COMPUTER."
110 PRINT A$
120 PRINT LEN(A$)
RUN
```

SEG$

The SEG$ (for "segment") statement lifts a segment of a string expression, called a *substring,* and prints it on the screen. For the sample program to work properly, line 100 must be reproduced exactly as printed, for example, two spaces separate the period and the I.

A$ is the string expression. When the computer executes line 110, it begins by locating the 15th character of A$ ("I") and counts the next 27 characters ("I" through "."). When this program is executed the computer prints the designated segment of the original string expression, then the entire string expression.

```
NEW
100 A$="HELLO THERE.  I AM THE TI-99/4A COMPUTER."
110 PRINT SEG$(A$,15,27)
120 PRINT A$
RUN

I AM THE TI-99/4A COMPUTER.
HELLO THERE. I AM THE TI-99/4A COMPUTER.
```

This program provides an example of how LEN and SEG$ can be used together. The five numbers input at line 110 print on the screen so that the number to the left of the decimal point is always aligned in the farthest right column. This type of alignment is called "right justification."

```
NEW
100 PRINT "INPUT FIVE NON-DECIMAL NUMBERS.
(PRESS ENTER AFTER EACH NUMBER.)"
110 FOR I=1 TO 5
120 INPUT B(I)
130 C$(I)=STR$(B(I))
140 NEXT I
150 FOR I=1 TO 5
160 PRINT B(I); TAB(27-LEN(C$(I)));C$(I)
170 NEXT I
180 END
RUN
```

DIM (Arrays)

The DIM statement (for "dimension") is used to reserve memory space for both numeric and string arrays. An ***array*** is a collection or set of variables. Each variable in an array has a ***subscript,*** a numeric expression that defines the position of each subscript in the array. An example of an array with 20 elements is: L(1), L(2), L(3),.....L(20). L is the array name and the numbers in parentheses are the subscripts.

The following program illustrates the use of the DIM statement and an array.

```
NEW
100 DIM N$(15)
```

Line 100 is the DIM statement. This line tells the computer to reserve 15 memory locations for the array (in this case, a list of words) represented by the string variable N$.

```
110 FOR X=1 TO 15
120 INPUT "ENTER A WORD ":N$(X)
130 NEXT X
```

Lines 110-130 create a FOR-NEXT loop with 15 iterations, or repetitions. Each iteration asks for input (ENTER A WORD) to be assigned to the array N$. For the first iteration, the value of X is 1. The first word entered is assigned to the variable N$(1). For the second iteration, the value of X is 2. The word entered is assigned to the variable N$(2). The process continues a total of 15 times.

```
140 CALL CLEAR
```

Line 140 clears the screen.

```
150 PRINT "YOUR LIST BACKWARDS IS:"
```

Line 150 prints a message.

```
160 FOR Q=15 TO 1 STEP -1
170 PRINT N$(Q)
180 NEXT Q
```

Lines 160-180 create a FOR-NEXT loop with a STEP. The normal default increment of a FOR-NEXT loop is one. To increment other than by one, a STEP function must be included in the FOR statement. In this case, STEP −1 causes the FOR-NEXT loop to decrement by one rather than increment by one. Simply stated, this loop prints the array backwards.

```
190 END
```

The subscript is Q, not X. Note that it is the value of Q or X that the computer uses to build or print arrays.

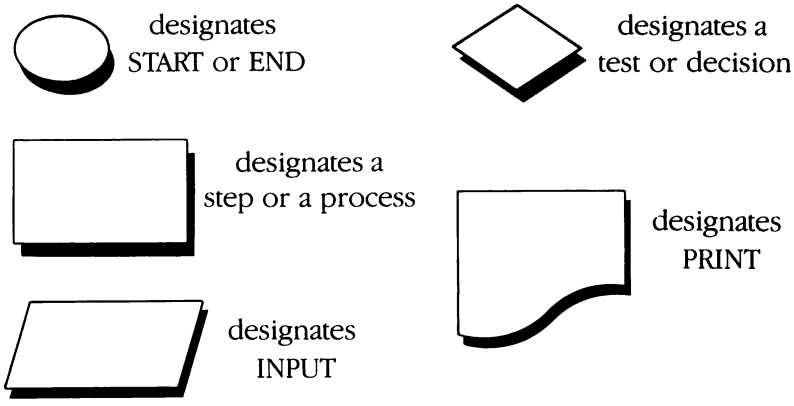
```
RUN
```

This program demonstrates the use of a one-dimensional array. For a detailed explanation of two- and three-dimensional arrays and arrays containing fewer than 12 elements, see *User's Reference Guide* (pages II-108–II-111).

FLOWCHARTING

As programs you develop grow in length, it may become more difficult to write them directly into the computer. **It is helpful to analyze the problem first, then draw a master plan, or *flowchart,* to lay out each step in the program.** A flowchart is a graphic representation of the procedures required to solve a problem.

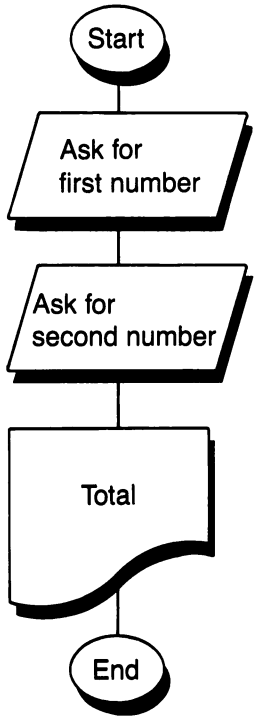
The common symbols used in flowcharting are:



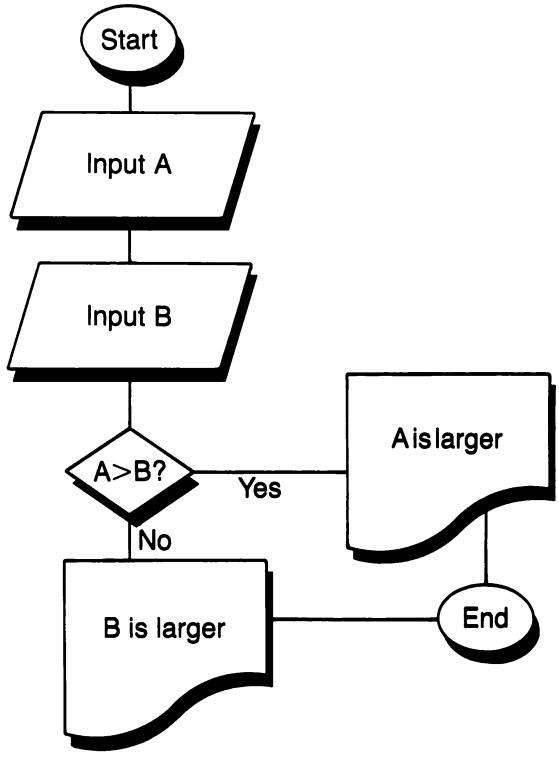
The two flowcharts presented here are the graphic representations of the programs listed below them.

Follow all the possible paths in the flowcharts mapping the two simple programs listed below.*

FLOWCHART 1:



FLOWCHART 2:




```
10 CALL CLEAR
20 "FIRST NUMBER? ":A
30  INPUT "SECOND
NUMBER? ":B
40 PRINT A+B
50 END
```

```
10 INPUT "ENTER A NUMBER
FOR A: ":A
20 INPUT "ENTER A NUMBER
FOR B: ":B
30 IF A>B THEN 60
40 PRINT "B IS LARGER"
50 GOTO 70
60 PRINT "A IS LARGER"
70 END
```

*For the purpose of these programs, A will not equal B.

CALL KEY

The CALL KEY statement allows you to put one character directly into a program from the keyboard. It eliminates the need to press ENTER. The following program illustrates the use of CALL KEY.

When this program is run, nothing seems to happen. The program is waiting for input from the keyboard for the letter K. Press K to see the results of this program. To stop program execution, press FCTN 4 (CLEAR).

For a complete explanation of CALL KEY consult the *User's Reference Guide* (pages II-87-II-89).

```
NEW
100 CALL KEY(0,K,S)
```

Line 100 is the CALL KEY statement. The O specifies the standard keyboard as the input device. The K is the return variable, i.e. the key that is pressed. The S is a status variable. If a key has not been pressed, the value of S is O.

```
110 IF S = 0 THEN 100
```

Line 110 tests the value of S. If a key has not been pressed, then the program loops back to line 100. If a key has been pressed, the computer executes line 120.

```
120 PRINT "YOU PRESSED ";CHR$(K)
130 GOTO 100
RUN
```

PROGRAMMING CHALLENGES

If you've understood programming to this point, you may be ready for a new challenge. Several problems are stated on the following pages. Read each problem carefully, making sure you fully understand it. Then decide what your program will need to do to solve the problem. Be sure to check carefully for errors.

When finished, run your program and correct any bugs you may encounter. Then refer to pages 35-36 (Answers to Programming Challenges) to compare your program with the sample given. Don't worry if your program doesn't look exactly like the example. Achieving the desired results is all that matters.

PART I

CONVERSION TABLES

LENGTH							
1 Kilometer	=	.621	Miles	**1 Mile (5,280 ft.)**	=	1.609	Kilometers
1 Meter	=	1.094	Yards	**1 Yard (3 ft.)**	=	.914	Meters
1 Centimeter	=	.394	Inch	**1 Foot (12 in.)**	=	30.480	Centimeters
1 Meter	=	3.281	Feet	**1 Inch**	=	2.540	Centimeters
CAPACITY							
1 Kiloliter	=	264.2	Gallons	**1 Gallon (4 qts.)**	=	3.785	Liters
1 Liter	=	1.06	Quarts	**1 Quart**	=	.946	Liters
WEIGHT							
1 Metric Ton	=	1.1	U.S. Tons	**1 U.S. Ton (2000 lbs.)**	=	.907	Metric Ton
1 Kilogram	=	2.2	Pounds	**1 Pound (16 oz.)**	=	.454	Kilograms
1 Gram	=	.04	Ounces	**1 Ounce**	=	28.35	Grams

Many of these conversion factors are rounded-off for ease of use.

PROGRAM I-A: Given any amount of gasoline in liters, write a program that will convert it to U.S. gallons.

PROGRAM I-B: Write a program that will convert your height and weight to metric measures (meters and kilograms).

PROGRAM I-C: Write a program that will convert the number of miles entered to yards, feet, inches, kilometers, meters, and centimeters, displaying all the answers on the screen.

PART II

PROGRAM II-A: Write a program that will print your name and address on the screen.

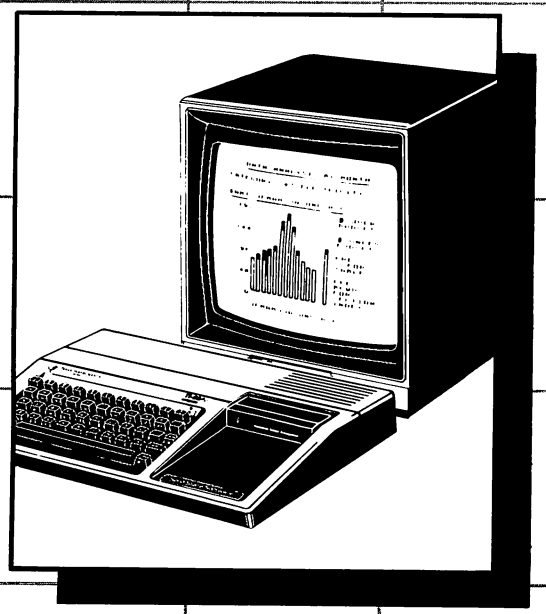
PROGRAM II-B: Write a program that will print your name and address on the screen the number of times specified by a user.

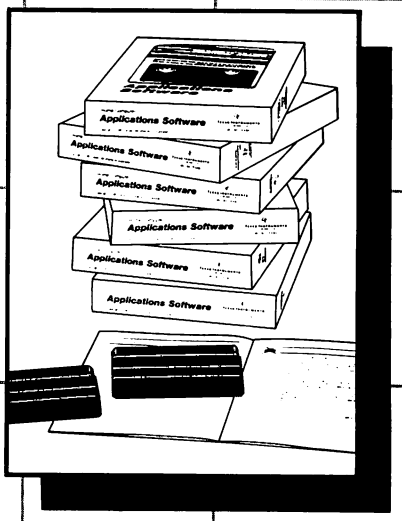
PROGRAM II-C: Write a program that will print two columns of your name and address, and print them as many times as specified by a user. (Name or address lines must not exceed 13 characters in length to fit on the screen.)

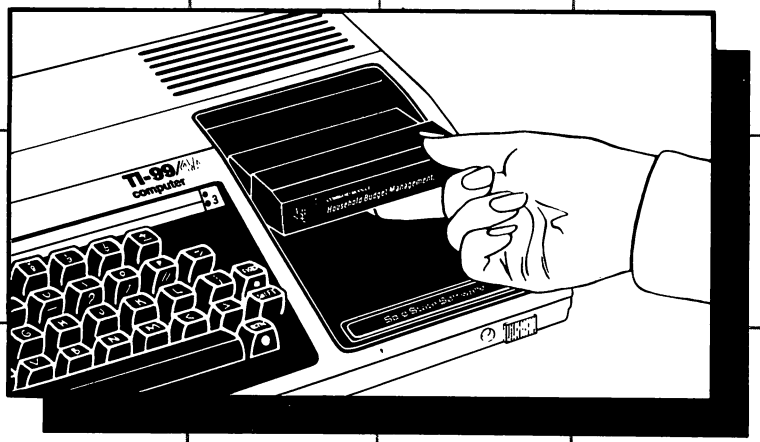
PART III

PROGRAM III-A: Write a program that will 1) display foreground color, background color and screen colors, 2) discuss the color appearing on the screen as they appear, and 3) incorporate appropriate sound or music.

APPENDIX




Applications Software
Applications Software
Applications Software
Applications Software
Applications Software
Applications Software


TI-99/4A
computer
Household Budget Management

SHORTCUTS AND EDITING FEATURES

Programs being entered into the computer or programs already written and loaded into the computer can be edited for corrections or modified to change the output of a program.

The TI-99/4A computer has several functions which can shortcut program writing and editing time. In becoming familiar with these functions, first review the keyboard functions listed on the inside front cover of this book.

To clear the screen of unwanted clutter:

CALL CLEAR
(ENTER)

Clears the screen, but not the memory.

To recall a program from memory:

LIST
(ENTER)

Displays program in memory on the screen.

To execute a program in memory:

RUN
(ENTER)

Causes the program in memory to execute.

To stop program execution:

FCTN 4 (CLEAR)

Program will cease running and print a BREAKPOINT message, communicating the line at which the program was stopped.

```
*BREAKPOINT AT 5
```

Before starting a new program, and before leaving a programming session, the memory should be cleared. To erase a current program from memory:

NEW
(ENTER)

NEW will clear the computer's memory and return to the TI BASIC READY screen.

BYE
(ENTER)

BYE will clear the computer's memory and return to the TI Master Title Screen, leaving TI BASIC. Using BYE is the preferred method of ending a programming session before turning off the computer system.

EDITING BY LINE

If an error is made while typing a program line, use the left arrow key (FCTN S) to backspace to the error. Correct the error by typing over it, then press ENTER, or press FCTN 3 (ERASE) to erase the entire line, then type the line (including the line number) correctly, and press ENTER. Use the space bar, FCTN 1 (DEL), FCTN 2 (INS), FCTN S (◀), FCTN D (▶), to make corrections. Refer to the following sections for more information on how to use these features.

If an error is detected in a program line already in memory, follow the directions given below.

To delete or change the contents of a program line:

EDIT 10
(ENTER)

Type EDIT 10 (this example is from the first program line in the INPUT section) and press ENTER. Line 10 and its contents appear on the screen.

OR

10
(FCTN X)

Optionally, type the desired line number and press FCTN X. Line 10 and its contents appear on the screen.

Using either of these methods brings the required line to the screen for alterations. Use the space bar, FCTN 1 (DEL), FCTN 2 (INS), FCTN S (◀), FCTN D (▶), to make corrections. Refer to the following sections for more information on how to use these features.

10 PRINT "THANK YOU FOR YOUR INPUT, JAMIE."

The flashing cursor appears on the first character of the statement following the line number.

If no change is desired in this line:

Press ENTER.

To change the contents of this line:

Use the arrow keys (FCTN S, FCTN D) to move the cursor to the point where change is desired and type over whatever is to be changed. Use the space bar to erase excess characters. Press ENTER to enter the line into the computer's memory.

To erase the contents of this line and enter other information:

Press FCTN 3 (ERASE), then type the information desired. Optionally, the DELete and INSert functions can also be employed (see the section below on DELete and INSert).

To erase this entire line from the program:

Press FCTN 3 (ERASE), then press ENTER. Optionally, instead of typing EDIT 10 and pressing ENTER to call line 10 to the screen, type 10, press ENTER, and the entire line will be erased from the computer's memory.

To remain in Edit Mode and view or edit other lines:

Press FCTN X (▼) to view lines in ascending order, or FCTN E (▲) to view lines in descending order.

To leave Edit Mode and effect changes made while in Edit Mode:

Press ENTER.

To leave Edit Mode and not effect changes made while in Edit mode:

Press FCTN 4 (CLEAR).

DELETE AND INSERT

To delete one or more characters within a line:

Move the cursor using the arrow keys (FCTN S, FCTN D). Place the cursor on the character to be deleted. Press FCTN 1 (DEL) once for each character to be removed. The character(s) will disappear and adjustment will be made automatically for the space remaining after the character(s) are deleted. Use an arrow key or press ENTER to exit the DELete function.

To insert one or more characters within a line:

Move the cursor using the arrow keys (FCTN S, FCTN D) on the character appearing to the right of the point where insertion is to occur. Press FCTN 2 (INS), release both keys, and type the desired character(s) to be added. Adjustment will automatically be made to provide space for the insertion of the character(s). Use an arrow key or press ENTER to exit the INSert function.

AUTOMATIC NUMBERING

The computer can be instructed to automatically provide numbers for program lines by use of Number Mode.

For automatic line numbering:

NUM
(ENTER)

NUM should be input before beginning a program. NUM will start line numbering at 100 and increase by increments of 10.

NUM 10
(ENTER)

NUM X will start line numbering at X and increase by increments of 10. In this case, numbering will begin with line 10 and continue in increments of 10.

NUM 5,5
(ENTER)

NUM X,Y will start line numbering at line X and increase by increments of Y. In this case, numbering will begin at line 5 and increase by increments of 5.

To insert automatically-numbered lines in an existing program:

Use NUM X,Y, X represents the point at which numbering should begin, and Y represents the incrementation.

To terminate automatic line numbering:

Press ENTER twice, first to enter your last valid program line, then again immediately after the next generated line number is displayed. The empty program line will not appear in the program.

RESEQUENCING LINE NUMBERS

If program line numbers are not evenly incremented after editing, the program line numbers can be automatically renumbered.

To resequence program line numbers by 10's starting with 100:

RES
(ENTER)
LIST
(ENTER)

Type RES, press ENTER, then LIST the program. The program lines will be renumbered automatically by 10's beginning with 100, including line numbers referenced in any GOTO statement.

To resequence program line numbers by 10's starting with 10:

RES 10
(ENTER)
LIST
(ENTER)

Type RES 10, press ENTER, then LIST the program. Resequencing begins with line 10. Line numbering in increments of 10 is standard for programs. It allows later insertion of program steps at specific points in a program.

To resequence program line numbers by X with increments of Y:

RES 5,5
(ENTER)
LIST
(ENTER)

Resequencing will begin with X and increase by Y, or any numbers inserted into this format. In the example, resequencing will begin with line number 5, followed by line 10, 15, 20 and so on.

NOTES ON LIST, RUN, AND TRACE

It is often desirable to view program lines in memory, especially after several edits have been performed.

To view a program in memory:

LIST
(ENTER)

Displays the program in memory.

If a program is more than 24 lines in length (a full screen), it is sometimes desirable to view only certain portions of the program at one time.

LIST 100-200
(ENTER)

Lists lines 100-200.

LIST 200-
(ENTER)

Lists the program beginning with line 200.

LIST -300
(ENTER)

Lists the program from the beginning through line 300.

The LIST command can also access peripherals.

LIST RS232
(ENTER)

Causes a program in memory to print out on a printer.

Typing RUN and pressing ENTER causes a program to execute beginning with the first program line. When editing a program, it is often desirable to stop a program at a certain point during execution and restart it from that point, or from another point, to access certain areas.

FCTN 4 (CLEAR)

Stops program execution.

CONTINUE
(ENTER)

Restarts program execution from the line at which the program was stopped.

RUN 300
(ENTER)

Causes a program to begin running from the line specified.

The TRACE command is another feature that can aid in the editing process. When TRACE is entered, and a program is run or continued, program line numbers are printed on the screen just before they are executed in the program.

TRACE
(ENTER)

Causes the TRACE feature to be activated.

RUN
(ENTER)

Executes the program. CONTINUE may also be used when applicable.

UNTRACE
(ENTER)

Causes the TRACE feature to be deactivated. NEW or BYE will also exit the TRACE command.

RUN

COLOR CODE CHART

COLOR	CODE	COLOR	CODE	COLOR	CODE
Transparent	1	Dark Red	7	Light Yellow	12
Black	2	Cyan	8	Dark Green	13
Medium Green	3	Medium Red	9	Magenta	14
Light Green	4	Light Red	10	Gray	15
Dark Blue	5	Dark Yellow	11	White	16
Light Blue	6				

MUSICAL TONE FREQUENCIES

FREQUENCY	NOTE	FREQUENCY	NOTE	FREQUENCY	NOTE
110	A	294	D	740	F#, G^{b}
117	A#, B^{b}	311	D#, E^{b}	784	G
123	B	330	E	831	G#, A^{b}
131	C (low C)	349	F	880	A (above high C)
139	C#, D^{b}	370	F#, G^{b}	932	A#, B^{b}
147	D	392	G	988	B
156	D#, E^{b}	415	G#, A^{b}	1047	C
165	E	440	A (above middle C)	1109	C#, D^{b}
175	F	466	A#, B^{b}	1175	D
185	F#, G^{b}	494	B	1245	D#, E^{b}
196	G	523	C (high C)	1319	E
208	G#, A^{b}	554	C#, D^{b}	1397	F
220	A (below middle C)	587	D	1480	F#, G^{b}
233	A#, B^{b}	622	D#, E^{b}	1568	G
247	B	659	E	1661	G#, A^{b}
262	C (middle C)	698	F	1760	A
277	C#, D^{b}				

CHARACTER SET CODES CHART

SET #1		SET #2		SET #3		SET #4		SET #5		SET #6	
Code	Character	Code	Character	Code	Character	Code	Character	Code	Character	Code	Character
32	(space)	40	(	48	0	56	8	64	@	72	H
33	!	41	)	49	1	57	9	65	A	73	I
34	"	42	*	50	2	58	:	66	B	74	J
35	#	43	+	51	3	59	;	67	C	75	K
36	$	44	,	52	4	60	<	68	D	76	L
37	%	45	-	53	5	61	=	69	E	77	M
38	&	46	.	54	6	62	>	70	F	78	N
39	'	47	/	55	7	63	?	71	G	79	O

SET #7		SET #8		SET #9		SET #10		SET #11		SET #12	
Code	Character	Code	Character	Code	Character	Code	Character	Code	Character	Code	Character
80	P	88	X	96	`	104	H	112	P	120	X
81	Q	89	Y	97	A	105	I	113	Q	121	Y
82	R	90	Z	98	B	106	J	114	R	122	Z
83	S	91	[	99	C	107	K	115	S	123	{
84	T	92	\	100	D	108	L	116	T	124	¦
85	U	93	]	101	E	109	M	117	U	125	}
86	V	94	^	102	F	110	N	118	V	126	~
87	W	95	_	103	G	111	O	119	W	127	DEL

ANSWERS TO PROGRAMMING CHALLENGES

Note: REM (for "remark") allows a programmer to include comments within a program. Comments appearing on a REM line appear only in the listing of a program and not on the screen when the program is executed.

PART I

Program I-A:

```
100 REM LITERS TO GALLONS
110 CALL CLEAR
120 INPUT "ENTER THE NUMBER OF LITERS YOU
WANT CONVERTED TO GALLONS: ":L
130 LET G=(L*1.06)/4
140 CALL CLEAR
150 PRINT L;" LITERS ARE EQUIVALENT TO "
;G;" U.S. GALLONS"
160 END
```

Program I-B:

```
90 REM HEIGHT & WEIGHT
110 CALL CLEAR
120 INPUT "HOW MANY FEET TALL ARE YOU? (ENTER
NUMBER OF FEET ONLY): ":F
130 INPUT "AND HOW MANY INCHES? (ENTER NUMBER
OF INCHES IN EXCESS OF FEET): ":I
140 INPUT "HOW MANY POUNDS DO YOU WEIGH? ":P
150 LET MH=(82.54)/100
160 LET MW=P*.454
170 CALL CLEAR
180 PRINT "YOU ARE";MH;" METERS HIGH AND WEIGH"
;MW; "KILOGRAMS"
190 END
```

Program I-C:

```
100 REM MILES TO EVERYTHING
110 CALL CLEAR
120 INPUT "ENTER THE NUMBER OF MILES YOU
WANT CONVERTED: ":M
130 PRINT :::::
140 LET YARDS=M*1760
150 LET FEET=M*5280
160 LET INCH=M*5280*12
170 LET KM=M*1.609
180 LET METER=M*1609
190 LET CM=M*160900
200 PRINT "IN";M;" MILES THERE ARE:"::
210 PRINT YARDS;" YARDS"
220 PRINT FEET;" FEET"
230 PRINT INCH;" INCHES"
240 PRINT KM;" KILOMETERS"
250 PRINT METER;" METERS, AND"
260 PRINT CM;" CENTIMETERS"
270 END
```

ANSWERS TO PROGRAMMING CHALLENGES

PART II

Program II-A:

```
10 CALL CLEAR
20 PRINT "William Smith"
30 PRINT "10 Any Street"
40 PRINT "Anywhere, FL 00038"
50 END
```

Program II-B:

```
10 CALL CLEAR
20 INPUT "HOW MANY TIMES DO YOU
 WANT YOUR ADDRESS PRINTED? ":X
30 CALL CLEAR
40 FOR TIMES=1 TO X
50 PRINT "William Smith"
60 PRINT "10 Any Street"
70 PRINT "Anywhere, FL   00038"
80 PRINT :::
90 NEXT TIMES
100 END
```

Program II-C:

```
10 CALL CLEAR
20 INPUT "HOW MANY TIMES DO YOU
WANT YOUR ADDRESS PRINTED? ":X
30 CALL CLEAR
40 FOR TIMES=1 TO X
50 N$="William Smith"
60 A$="10 Any Street"
70 CS$="Anywhere, FL"
80 Z$="                 00038"
90 PRINT N$,N$
100 PRINT A$,A$
110 PRINT CS$,CS$
120 PRINT Z$,Z$
130 PRINT :::
140 NEXT TIMES
150 END
```

PART III

```
90 REM DEMO
100 OPEN #1: "SPEECH", OUTPUT
110 CALL CLEAR
120 INPUT "SCREEN COLOR: ":S
130 T=S
140 GOSUB 500
150 S$=T$
160 INPUT "FOREGROUND: ":F
170 T=F
180 GOSUB 500
190 F$=T$
200 INPUT "BACKGROUND: ":B
210 T=B
220 GOSUB 500
230 B$=T$
240 CALL CLEAR
250 CALL SCREEN(S)
260 CALL COLOR(2,F,B)
270 CALL HCHAR (12,3,42,28)
280 PRINT #1:"THE SCREEN COLOR IS"
290 PRINT #1:S$
300 PRINT #1:"THE FOUR+GROUND COLOR IS"
310 PRINT #1:F$
320 PRINT #1:"THE BACKGROUND COLOR IS"
330 PRINT #1:B$
340 CALL SOUND (75, 262,0,330,0,392,0)
350 CALL SOUND (1000,262,0,330,0,392,0)
360 GOTO 120
500 ON T GOTO 510,520,530,540,550, 560,570,
580,590,600,610,620,630,640,650,660
510 T$="TRANSPARENT"
511 RETURN
520 T$="BLACK"
521 RETURN
530 T$="MEEDEEUM GREEN"
531 RETURN
540 T$="LIGHT GREEN"
541 RETURN
550 T$="DARK BLUE"
551 RETURN
560 T$="LIGHT BLUE"
561 RETURN
570 T$="DARK RED"
571 RETURN
580 T$="SIGH+ANN"
581 RETURN
590 T$="MEEDEEUM RED"
591 RETURN
600 T$="LIGHT RED"
601 RETURN
610 T$="DARK YELLOW"
611 RETURN
620 T$="LIGHT YELLOW"
621 RETURN
630 T$="DARK GREEN"
631 RETURN
640 T$="MAGENTA"
641 RETURN
650 T$="GRAY"
651 RETURN
660 T$="WHITE"
661 RETURN
```

MORE PROGRAMS

Programs A and B are similar. They both instruct the computer to add numbers from 1 to 100 and print the sum on the screen. Program A is different from Program B since it has the computer add and print the sum of all odd numbers from 1 to 1000. Program B adds or prints the sum of all even numbers from 2 to 1000.

To watch the computer as it works a problem, add line 125 to either or both of the programs.

Program A

```
100 CALL CLEAR
110 FOR X=1 TO 1000 STEP 2
120 SUM=SUM+X
130 NEXT X
140 PRINT "THE SUM OF ALL ODD
NUMBERS FROM 1 TO 1000 IS ":SUM
150 END
125 PRINT X;SUM
```

Program B

```
100 CALL CLEAR
110 FOR X=2 TO 1000 STEP 2
120 SUM=SUM+X
130 NEXT X
140 PRINT "THE SUM OF ALL EVEN
NUMBERS IS";SUM
150 END
```

Program C allows you to try different combinations of screen, character, and background colors by entering the desired color code numbers (1-16) and using INPUT statements (lines 20-40). Refer to the Color Code Chart on page 34 for code numbers.

Program C

```
10 CALL CLEAR
20 INPUT "SELECT A SCREEN COLOR
NUMBER FROM 1 TO 16: ":S
30 INPUT "SELECT A FOREGROUND COLOR
NUMBER FROM 1 TO 16: ":F
40 INPUT "SELECT A BACKGROUND COLOR
NUMBER FROM 1 TO 16: ":B
50 CALL CLEAR
60 CALL SCREEN(S)
70 CALL COLOR (2,F,B)
80 CALL HCHAR(12,3,42,28)
90 GOTO 20
```

Program D produces randomly-generated stars on a black screen.

Program D

```
100 CALL CLEAR
110 CALL SCREEN(2)
120 CALL COLOR (2,26,1)
130 RANDOMIZE
140 ROW=INT(RND*24)+1
150 COL=INT(RND*32)+1
160 CALL HCHAR(ROW,COL,42)
161 FOR DELAY=1 TO 200
162 NEXT DELAY
170 GOTO 140
```

Program E is an interesting graphics program called Blinds. Venetian blinds open, close, and change color.

Program E

```
10 CALL CLEAR
20 A$="FFFFFFFFFFFFFFFF"
30 FOR I=1 TO LEN(A$)STEP 2
40 CALL CHAR(32,SEG$,(A$,I,16))
50 NEXT I
60 CALL SCREEN(INT (RND*14+2))
70 GOTO 30
```

MORE PROGRAMS

Program F computes 5% tax on a given amount of money. The input line (line 110) asks for the amount or price of an item. The answer prints the amount of tax on the screen (line 130) and the total of tax plus the original amount (line 140).

Program F

```
100 CALL CLEAR
110 INPUT "PRICE =":P
120 TAX=P*.05
130 PRINT "TAX ON";P; "IS";TAX::
140 PRINT "TOTAL COST IS";P+TAX
```

Program G is similar to Program F but has been modified to hold the answer on the screen for about 5 seconds (lines 150-160) before returning to the beginning of the program and asking for another amount to figure.

Program G

```
100 CALL CLEAR
110 INPUT "PRICE =":P
120 TAX=P*.05
130 PRINT "TAX ON";P;"IS";TAX::
140 PRINT "TOTAL COST IS";P+TAX
150 FOR DELAY=1 TO 1000
160 NEXT DELAY
170 GOTO 100
```

Program H performs the same function as Program F, but contains a formula to round the tax amount to two decimal places (lines 121-124).

Program H

```
100 CALL CLEAR
110 INPUT "PRICE =":P
120 TAX=P*.05
121 TAX=TAX*100
122 TAX=TAX+.5
123 TAX=INT(TAX)
124 TAX=TAX/100
130 PRINT "TAX ON";P;"IS";TAX::
140 PRINT "TOTAL COST IS";P+TAX
```

Program I performs the same function as Program H, but lines 121-124 of Program H have been included in one program statement (line 120).

Program I

```
100 CALL CLEAR
110 INPUT "PRICE =":P
120 TAX=(INT(((P*.05)*100)+.5))/100
130 PRINT "TAX ON";P;"IS";TAX::
140 PRINT "TOTAL COST IS";P+TAX
```

Program J allows a name of up to 28 characters in length to be entered, and then centers that name on the screen.

Program J

```
100 CALL CLEAR
110 PRINT "INPUT COMPANY NAME:"::
120 INPUT N$
130 X=LEN(N$)
140 IF X>28 THEN 100
150 COL=INT((30-X)/2)
160 CALL CLEAR
170 PRINT TAB(COL);N$::::::::::
180 GOTO 110
```

For Program K to run, the Terminal Emulator II cartridge and the Speech Synthesizer must both be attached. Program K demonstrates how speech can be used in a TI BASIC program, as well as changing the quality of the voice (line 130).

Program K

```
100 OPEN #1:"SPEECH",OUTPUT
110 PRINT #1:"HELLO"
120 PRINT #1:"CONGRATULATIONS. YOU
HAVE SUCCESSFULLY ACTUHVAITED THE
NINE TEE NINE 4 A SPEECH UNIT."
130 PRINT #1:"//20 100"
140 PRINT #1:"IS MY VOICE HIGHER NOW?"
300 CLOSE #1
```

MORE PROGRAMS

Program L requires an RS-232 and a printer. To run this program without these peripherals, delete lines 100, 260, 280, 350, 370, and 400. This program converts miles to kilometers and kilometers to miles. It allows the user to select the type of conversion desired from a menu (lines 120-180). QUIT is also an option on the menu (choice 3). If QUIT is selected, the program will end.

If choice 1 is selected, the program goes to line 220, where an input statement asks for the number of miles to be converted. The answer prints on the screen and on a printer (if a printer is attached).

If choice 2 is selected, the program branches to line 310 where the input statement asks for the numbers of kilometers to be converted. The answer prints on the screen and on a printer (if attached).

After each answer is displayed, the selection menu appears on the screen and "waits" for another selection.

Program M creates a graphic title screen for Program L. Program M is numbered by fives to facilitate adding it to Program L. If you add Program M to Program L, first remove line 65.

Program L

```
100 OPEN #1:"RS232"
110 CALL CLEAR
120 PRINT "PRESS 1"
130 PRINT "FOR MILES TO KILOMETERS"::
140 PRINT "PRESS 2"
150 PRINT "FOR KILOMETERS TO MILES"::
160 PRINT "PRESS 3"
170 PRINT "TO QUIT"::
180 INPUT "CHOICE? ":CHOICE
190 IF CHOICE<1 THEN 110
200 IF CHOICE >3 THEN 110
210 ON CHOICE GOTO 220,310,400
220 INPUT "MILES = "M
230 K=M*1.609
240 CALL CLEAR
250 PRINT M;"MILES ARE EQUAL TO"::
260 PRINT #1:M;"MILES ARE EQUAL TO"
270 PRINT K;"KILOMETERS"::
280 PRINT #1:K;"KILOMETERS"::
290 GOSUB 120
300 RETURN
310 INPUT "KILOMETERS = ":K
320 M=K/1.609
330 CALL CLEAR
340 PRINT K;"KILOMETERS ARE EQUAL
TO"::
350 PRINT #1:K;"KILOMETERS ARE EQUAL
TO"
360 PRINT M;"MILES"::
370 PRINT #1:M;"MILES"::
380 GOSUB 120
390 RETURN
400 CLOSE #1
410 END
```

Program M

```
  5 CALL CLEAR
 10 CALL SCREEN(5)
 15 CALL HCHAR(22,2,42,30)
 20 PRINT TAB(9);"U.S. TO METRIC"
 25 PRINT TAB(10);"CONVERSIONS"::
 30 CALL HCHAR(24,2,42,30)
 35 PRINT ::::::::
 40 CALL VCHAR(11,2,42,4)
 45 CALL VCHAR(11,31,42,4)
 50 FOR CHARSET=1 TO 16
 55 CALL COLOR(CHARSET,16,1)
 60 NEXT CHARSET
 65 GOTO 65
```

MORE PROGRAMS

Program N was designed for a fictional Mrs. Wilson, who runs a tailoring service from her home. When this program is entered and run, the computer becomes a specialized type of "cash register." When one or more code numbers are entered (line 140) the program first tests to see if a wrong number has been entered (line 150), then asks if more codes are to be input. When N is pressed at line 160, the program then searches the DATA statements (lines 200-250) for the data corresponding to the codes entered. The computer then formats the requested data on the screen (lines 350-390) and adds up the total price of services performed, including 4% sales tax (lines 420-460).

Mrs. Wilson's program can be modified to serve as a cash register for many types of home services by changing and adding DATA statements, adjusting state sales tax, etc.

Program N

```
100 CALL CLEAR
110 DIM SC(25)
120 GOSUB 480
130 FOR X=1 TO 25
140 INPUT "ENTER CODE NUMBER: ":SC(X)
150 IF SC(X) 5 THEN 140
160 INPUT "ARE THERE MORE?(Y?N) ":M$
170 PRINT
180 IF M$="N" THEN 200
190 NEXT X
200 DATA 1,CUFF PANTS, 3.49
210 DATA 2,ALT WAIST,4.95
220 DATA 3,HEM SKIRT,3.19
230 DATA 4,SEW BUTTON,1.05
240 DATA 5, MISC ALTER,4.99
250 DATA 999,FILLER,0
260 FOR Y=1 TO X
270 READ N,J$,COST
280 IF N=999 THEN 580
290 IF N=SC(Y) THEN 300 ELSE 270
300 SN$(Y)=J$
310 PR(Y)=COST
320 RESTORE
330 NEXT Y
340 CALL CLEAR
350 PRINT "CODE     DESCRIP     PRICE"
360 PRINT "-----------------------------------------"
370 FOR Y=1 TO X
380 PRINT SC(Y);TAB(7);SN$(Y);
TAB(19);PR(Y)
390 TOT=TOT+PR(Y)
400 NEXT Y
410 PRINT "-----------------------------------------"
420 PRINT "SUBTOTAL";TAB(19);TOT
430 TAX=INT(TOT*.04*100+.5)/100
440 PRINT "4% TAX";TAB(20);TAX
450 PRINT "-----------------------------------------"
460 PRINT "GRAND
TOTAL";TAB(19);TAX+TOT
470 END
480 PRINT
490 PRINT "DESCRIPTION", "CODE NUMBER"
500 PRINT "-----------------------------------------"
510 FOR Q=1 TO 5
520 READ A,T$,COST
530 PRINT T$,A
540 NEXT Q
550 PRINT
560 RESTORE 200
570 RETURN
580 PRINT "CODE";SC(Y); "NOT FOUND.":
```

Program O is also entitled "What's for Supper?" When a date is entered (line 250) the program randomly generates a week's menu based on the meals listed in the DATA statements (lines 160-190) and prints a week's menu on the screen. The meals listed in lines 160-190 can be easily changed to reflect personal favorites.

```
"TRY AGAIN."
590 RESTORE
600 PRINT :
610 GOTO 120
```

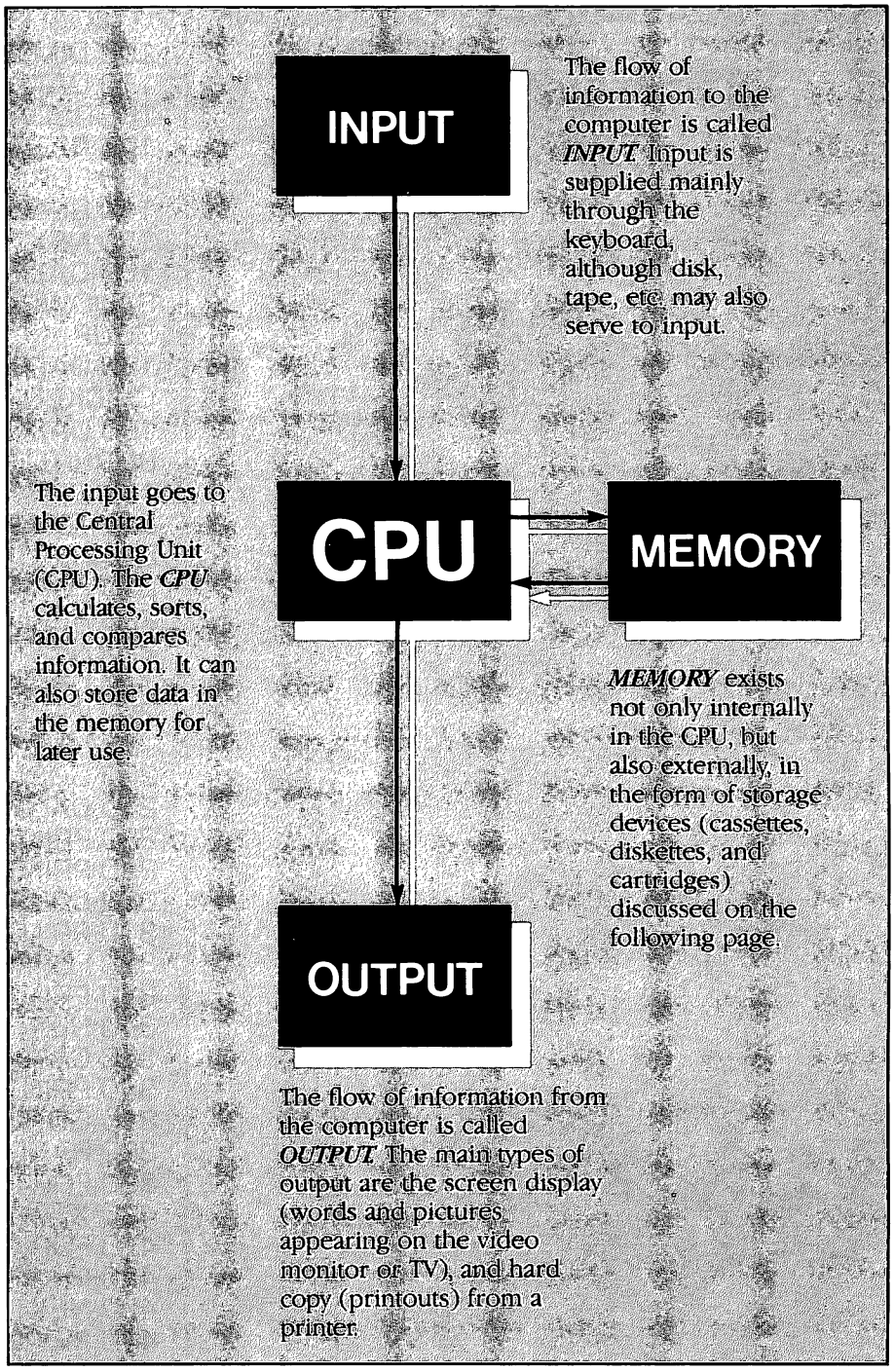
Program O

```
100 DIM DAY$(7),MEAL$(7),MENU$(26)
110 DATA MONDAY, TUESDAY, WEDNESDAY,
THURSDAY, FRIDAY, SATURDAY, SUNDAY
120 RESTORE 110
130 FOR I=1 TO 7
140 READ DAY$(I)
150 NEXT I
160 DATA HAMBURGERS, HOTDOGS,
SANDWICHES, PIZZA, COLD CUTS, SALAD,
SOUP
170 DATA FRIED CHICKEN, BAKED
CHICKEN, ROAST, STEAK, CHOPPED STEAK
180 DATA SPAGHETTI, TACOS, TAMALES,
BURRITOS, HAM, EGGS, GO OUT TO EAT
190 DATA CHILI, PORK CHOPS, FISH TV
DINNER, RIBS, PANCAKES, FRUIT
200 RESTORE 160
210 FOR I=1 TO 26
220 READ MENU$(I)
230 NEXT I
240 CALL CLEAR
250 INPUT "MONDAY (MM,DD,YY):
":MM,DD,YY
260 GOSUB 380
270 CALL CLEAR
280 PRINT "MENU FOR WEEK BEGINNING"
290 PRINT "MONDAY";MM;"/";DD;"/";YY::
300 CALL HCHAR(23,3,45,28)
310 PRINT
320 FOR I=1 TO 7
330 PRINT DAY$(I);TAB(13);MEAL$(I)::
340 NEXT I
350 CALL HCHAR(24,3,45,28)
360 PRINT ::
370 GOTO 250
380 RANDOMIZE MM+DD
390 FOR I=1 TO 7
400 X=INT(RND*26)+1
410 FOR C=1 TO I
420 IF MEAL$(C)=MENUS$(X) THEN 400
430 NEXT C
440 MEAL$(I)=MENUS$(X)
450 NEXT I
460 RETURN
```

HOW A COMPUTER WORKS

There are two types of memory in a computer system: Random Access Memory (RAM) and Read Only Memory (ROM). Information stored in RAM is not permanent. It can be transferred to storage media (disk, cassette), retrieved from storage media, changed, reordered, and erased. When power to the system is turned off, whatever information was stored in the RAM disappears. For example, a program entered from the keyboard or loaded from disk or cassette is stored in RAM. ROM is built-in, permanently stored memory which cannot be altered in any way. An example of ROM is a pre-programmed cartridge.

Computers "remember" and "calculate" through a series of switches. Each switch codes a zero (off) or a one (on). This coding provides the computer with information. Each switch is called a ***bit.*** Eight bits are called a ***byte.*** It takes one byte of information to encode a single character from the keyboard, for example, the letter A. The TI-99/4A alone has the capacity to hold approximately 16,000 bytes of information in memory. This is normally referred to as 16K RAM. TI-99/4A peripherals can expand the system to 48K RAM. With addition of the Mini-Memory cartridge, the system can grow to 52K RAM.



PROGRAM AND DATA STORAGE

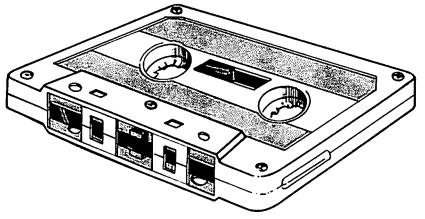
SOLID STATE CARTRIDGES

Equipment needed:

- Computer console
- TI Color Monitor or TV with adapter (Video Modulator)
- Cartridge (contains a printed circuit board with chips)

Cartridge software is an excellent way to take advantage of the wide assortment of pre-programmed software currently available. Most cartridges require no extra peripherals because they plug directly into the console. Cartridges, none of which are erasable with the exception of the Mini-Memory, are very easy to use. You can store personal data, temporarily or permanently, on the specialized Mini-Memory cartridge.

CASSETTES

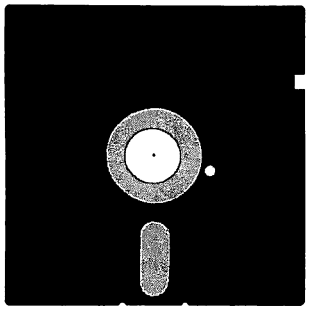


Equipment needed:

- Computer console
- TI Color Monitor or TV with adapter (Video Modulator)
- Compatible cassette recorder
- Cassette tape (high quality)
- Cassette cable

Using cassettes is an inexpensive way to store personal programs and data files, and to enjoy pre-programmed software. The small amount of equipment needed (recorder and tape) is not very expensive and, of course, can be used for other purposes. One or two cassette recorders can be controlled from the console. Sometimes using cassettes involves waiting and repeating the procedure to successfully load data into the computer. It is the preferred method of data storage if you choose to avoid the higher cost of a disk system.

DISKETTES



Equipment needed:

- Computer console
- TI Color Monitor or TV with adapter (Video Modulator)
- Disk Controller card with the Peripheral Expansion System (or TI Disk Controller peripheral)
- One or more TI Disk Memory Drives
- Diskettes (5¼ inch floppy disk)

Using the TI Disk Memory System is the fastest method of storing and retrieving personal data. Pre-programmed software is also available on disk. The same programs are usually available on both diskette and cassette. Though the equipment involved costs more than cassette, the speed and efficiency of using diskettes is well worth it. About 87,000 keystrokes of information can be stored on a single disk.

TIPS ON USING SOFTWARE

SOLID STATE CARTRIDGES

To Insert Solid State Cartridges

- Turn on your monitor or TV, then the console. The TI Master Title Screen should automatically display on your screen.
- Insert a solid state cartridge firmly into the slot to the right of the keyboard. The screen will go blank momentarily, then the TI Master Title Screen should reappear.
- Press any key to move to the next screen.
- Select the name of the cartridge you inserted usually by pressing the number 2 (the number 1 will always take you into TI BASIC).

To Remove Solid State Cartridges

- Return to the TI Master Title Screen by pressing FCTN and the " = " key, or by exiting according to instructions included in the cartridge's program.
- Remove the cartridge from its slot.

In Case of Difficulty

- If a cartridge does not seem to operate properly, reinsert the cartridge. If the screen locks or produces unusual displays, turn the console off and wait a few seconds before turning it back on.

CASSETTE SOFTWARE

Load and Save with Cassette

- Use only high-quality cassette tapes, sixty minutes or less in length.
- Connect a compatible cassette recorder to your computer with cassette cables. Insert the single plug at one end of the cable into the back of the console. The red-wired plug at the opposite end of the cable connects to the cassette microphone jack. The white-wired plug adjacent to it connects to the earphone or auxiliary cassette jack. It is not necessary to insert the black-wired remote plug into the cassette recorder (it can cause some cassette recorders to malfunction). If you are using dual cassette cables, insert the end marked CS1 into the cassette recorder.
- Make sure the ALPHA LOCK key is in locked (down) position.

Loading a Progam

- Make sure the cassette recorder is either plugged in or contains good batteries.
- Insert a cassette containing a program into the cassette deck.
- Turn on the monitor or TV, then the console.
- Select TI BASIC from the menu.
- When TI BASIC READY appears, type OLD CS1, then press ENTER.
- Follow the directions on the screen to "walk" through the loading procedure.
- When the DATA OK statement appears on the screen, wait for the flashing cursor to reappear.
- Type RUN and press ENTER. (It may take several seconds for the program to display its first screen on your video.)

Saving a Program

- Prepare a program to save by either writing or loading it from another source.
- Insert a blank cassette tape into the cassette recorder.
- Turn on the monitor or TV, then the console.
- Select TI BASIC.

- Type SAVE CS1, then press ENTER.
- Follow the instructions on the screen to "walk" through the SAVE procedure.
- When the SAVE procedure is complete, the question CHECK TAPE(Y OR N)? appears on the screen. Respond with "Y" to check the recording.

If You Have Difficulties

An error message reading "NO DATA FOUND" may be displayed. If this happens, check for the following problems and correct them if necessary:

- an unplugged cassette recorder or weak batteries
- improperly placed or loose connections
- cassette volume level (it should be around 8)
- cassette tone level (it should be around 6)
- an excessively long leader on your cassette tape (after you press the PLAY button on your recorder, wait a few seconds, then press ENTER)

If these common problems have been corrected but error messages continue, the cassette recording may be damaged, the cassette recorder may be incompatible with the TI-99/4A system, or improperly functioning equipment may be found.

DISKETTE SOFTWARE

To Load and Save from Diskette

- Follow the set-up procedures outlined in the Disk Memory System manual.
- Use standard 5¼-inch, soft-sectored, 40-track floppy diskettes.
- Make sure ALPHA LOCK is in the locked (down) position.

Loading a Program

- Turn on the Peripheral Expansion Box (or disk drive and controller), the monitor or TV, and finally the computer.
- Insert a diskette containing stored programs into the disk drive.
- Close the disk drive door.
- Select TI BASIC.
- When TI BASIC READY appears on the screen, type OLD DSK1.-------- (where -------- represents the name of the program) and press ENTER.
- When the flashing cursor reappears on the screen (this may take several seconds to occur), type RUN and press ENTER.

Saving a Program

- Initialize a blank diskette using the Disk Manager cartridge. Refer to the Disk Manager manual for instructions on initialization.
- Prepare a program to save by either writing or loading it from another source.
- Insert the initialized diskette into the disk drive unit.
- Close the disk drive unit door. Never open a disk drive door when the red light is on.
- Type SAVE DSK1.-------- (where -------- represents the name of the program) and press ENTER.
- Remove the diskette when the SAVE procedure is completed. For future reference, label the diskette with the name of the program(s) saved.

SOME FINAL TIPS

As a strict rule, keep all software (especially diskettes and cassettes) away from heat, magnetic fields (metal detectors, TVs, monitors, magnets), and static electricity.

Handle disks with care. Be careful not to soil the magnetic surface or to bend or warp the disks.

Reading the software and hardware instruction booklets, and keeping them nearby for quick reference, can save a great deal of time and aggravation.

AVAILABLE SOFTWARE PACKAGES

INFORMATION MANAGEMENT

CARTRIDGES

Home Financial Decisions

Household Budget Management (Cassette or Disk Data storage system required.)

Securities Analysis

Personal Record Keeping (Cassette or Disk Data storage system recommended.)

Tax/Investment Record Keeping (Disk system required.)

Personal Real Estate (Cassette or Disk Data storage system recommended.)

Personal Report Generator (Disk drive/controller or Cassette recorder/cable, TI Impact Printer/RS-232 Interface required.)

TI Writer* (Disk drive/controller, 32K Memory Expansion Unit, TI Impact Printer/RS-232 Interface required.)

Microsoft™ Multiplan[1]* (Disk drive/controller, 32K Memory Expansion Unit required. TI Impact Printer/RS-232 Interface recommended.)

Terminal Emulator II (Telephone Coupler and RS-232 Interface required for telecommunications. Speech Synthesizer required for text-to-speech.)

DISKS AND CASSETTES

Mailing List (Disk only)

Personal Finance Aids (Disk and Cassette)

Checkbook Manager (Disk only)

Business Aids Library

- Financial Management (TI Extended BASIC cartridge required. Disk only)
- Inventory Management (Disk only)
- Invoice Management (Personal Record Keeping or Statistics required. Disk only)
- Cash Management (TI Extended BASIC cartridge required. Disk only)
- Lease/Purchase Decisions (Disk and Cassette)

Personal Tax Plan[2] (Disk drive/controller, P-Code Card, and 32K Memory Expansion Unit required. TI Impact Printer/RS-232 Interface recommended. Disk only)

TI-Count Business Packages[3] (TI Extended BASIC cartridge, TI Impact Printer/RS-232 Interface required. 32K Memory Expansion and second Disk drive recommended. Disk only)

EDUCATION

CARTRIDGES

Early Learning Fun (For ages 3-6)

Beginning Grammar (For grade levels 2-5)

Number Magic (For ages 6 and up)

Video Graphs (For all ages)

Physical Fitness (For ages 13 and up)

Early Reading[4] (For beginning readers. Speech Synthesizer required.)

Reading Fun[4] (For grade level 2. Speech Synthesizer optional.)

Reading On[4] (For grade level 3.)

Reading Roundup[4] (For grade level 4.)

Reading Rally[4] (For grade level 5.)

Reading Flight[4] (For grade level 6.)

Addition/Subtraction 1[4] (For grade level 1. Speech Synthesizer recommended.)

Addition/Subtraction 2[4] (For grade levels 1-2. Speech Synthesizer recommended.)

Multiplication 1[4] (For grade levels 3-4. Speech Synthesizer recommended.)

Division 1[4] (For grades 3-5. Speech Synthesizer optional.)

Computer Math Games II & VI[5] (For grade levels 1-9.)

Developmental Learning Materials (DLM) Series:

Alien Addition (for addition skills)

Minus Mission (for subtraction skills)

Alligator Mix (for discrimination skills)

Meteor Multiplication (for multiplication skills)

Demolition Division (for division skills)

Dragon Mix (for discrimination skills)

Music Maker (Cassette or Disk Data storage system recommended.)

Weight Control and Nutrition (Cassette or Disk Data storage system recommended.)

TI LOGO II (32K Memory Expansion Unit required. TI Impact Printer/RS-232 Interface, Disk drive/controller, Cassette recorder/cable, all optional.)

Scholastic Spelling[6] (For grade levels 3-6. Speech Synthesizer required.)

Milliken Math Series (For grade levels K-8.)

- Addition
- Multiplication
- Integers
- Decimals
- Laws of Arithmetic
- Measurement Formulas
- Subtraction
- Division
- Fractions
- Percents
- Equations

DISKS AND CASSETTES

Music Skills Trainer (For ages 10 and up. Disk and Cassette)

Computer Music Box (For ages 10 and up. Disk and Cassette)

Market Simulation (Disk and Cassette)

Music Maker Demonstration (Music Maker cartridge required. Disk only)

Basketball Statistician (TI Extended BASIC cartridge required. Disk only)

Bridge Bidding I (Disk and Cassette)

Bridge Bidding II (Disk and Cassette)

Bridge Bidding III (Disk and Cassette)

Speak & Spell™ Program (Disk only)

Speak & Math™ Program (Disk and Cassette)

Spell Writer (Terminal Emulator II cartridge and Speech Synthesizer required. Disk and Cassette)

TI PILOT (Disk drive/controller, 32K Memory Expansion Unit, and P-Code Card required. Editor/Filer/Utilities disk required for UCSD P-System program development. Speech Synthesizer optional for speaking programs. Disk only)

Course Designer Authoring System* (TI Extended BASIC cartridge required. Speech Synthesizer, TI Impact Printer/RS-232 Interface, Video controller, all optional. Disk only)

SMU Electrical Engineering Library (Cartridge with Disk or Cassette available)

Text-to-Speech (English) (Disk drive/controller, Speech Synthesizer, 32K Memory Expansion Unit, TI Extended BASIC cartridge required. Disk only)

Touch Typing Tutor*

AVAILABLE SOFTWARE PACKAGES

PLATO® LEARNING CENTER*

Now, the power of the TI Home Computer makes PLATO courses on disks available to everyone. Choose from more than 450 programs in the PLATO curriculum. Here's all that's required to use PLATO courseware:

- TI-99/4A Home Computer
- TI Disk Memory System
- TI Memory Expansion
- PLATO Interpreter Solid State Cartridge
- PLATO Program Packages (your choice)

Parents interested in Basic Skills programs for their children receive additional help when they buy the PLATO Interpreter Solid State Cartridge, because the package includes:

- The Survey Disks, which ask your child questions to determine their strengths and weaknesses in reading, grammar, and math.
- The Parent's Questionnaire, which asks you questions that help you assess your child's academic skills.

BASIC SKILLS (GRADES 3-8)

Mathematics

- Basic Number Ideas
- Addition
- Subtraction
- Multiplication
- Division
- Fractions
- Decimals
- Ratio, Proportion, and Percent
- Geometry and Measurement

Reading

- Making New Words
- Understanding New Words
- Understanding What You Read
- Thinking About What You Read
- Judging What You Read

Grammar

- Parts of Speech
- Building and Using Sentences
- Spelling and Usage
- Capital Letters and Punctuation
- Writing Letters

HIGH SCHOOL SKILLS (GRADES 9-12)

Mathematics

- Basic Number Ideas
- Math Sentences in One Variable
- Math Sentences in Two Variables
- Geometry
- Measurement
- Special Topics

Reading

- Reading
- General Reading
- Prose Literature
- Poetry
- Drama

Writing

- Spelling and Punctuation
- Grammar
- Diction
- Sentence Structure
- Logic and Organization

Science

- Physics
- Chemistry
- Earth Science
- Biology

Social Studies

- Geography
- Economics
- Behavioral Science
- Political Science
- History

ENTERTAINMENT

CARTRIDGES

Parsec* (Speech Synthesizer and Wired Remote Controllers optional.)
Tombstone City: 21st Century (Wired Remote Controllers optional.)
TI Invaders (Wired Remote Controllers optional.)
Car Wars (Wired Remote Controllers optional.)
Alpiner* (Speech Synthesizer, Wired Remote Controllers optional.)
Othello[7]
Chisholm Trail (Wired Remote Controllers optional.)
Football
Video Games I (Wired Remote Controllers optional.)
Hunt the Wumpus (Wired Remote Controllers optional.)
Indoor Soccer (Wired Remote Controllers optional.)
Mind Challengers
A-Maze-Ing (Wired Remote Controllers optional.)
The Attack[8] (Wired Remote Controllers optional.)
Blasto[8] (Wired Remote Controllers optional.)
Blackjack and Poker (Wired Remote Controllers optional.)
Hustle[8] (Wired Remote Controllers optional.)
ZeroZap[8]
Hangman[8]
Connect Four[8]
Yahtzee[8]
Video Chess
E.T.[9], The Extra-Terrestrial (Speech Synthesizer recommended.)
Munch Man

DISKS AND CASSETTES

Tunnels of Doom (Cartridge with Disk or Cassette available)
Adventure International Series (Cartridge with The Pirate Adventure disk and cassette available.)

- Adventureland
- Voodoo Castle
- Strange Odyssey
- Pyramid of Doom
- Savage Island I & II
- Mission Impossible
- The Count
- Mystery Fun House
- Ghost Town
- The Golden Voyage

Mystery Melody (Disk and Cassette)
Oldies But Goodies–Games I (Disk and Cassette)
Oldies But Goodies–Games II (Disk and Cassette)
Saturday Night Bingo (Speech Synthesizer required. Disk and Cassette)
Draw Poker (TI Extended BASIC cartridge required. Disk and Cassette)

AVAILABLE SOFTWARE PACKAGES

COMPUTER PROGRAMMING

CARTRIDGES

Speech Editor (Speech Synthesizer required.)
Editor/Assembler (32K Memory Expansion Unit and Disk drive/controller required.)
Mini-Memory (Cassette recorder/cable recommended. May also be used with all other peripherals.)
TI Extended BASIC

DISKS AND CASSETTES

Pascal Development System (Peripheral Expansion System, 32K Memory Expansion, Disk drive/controller required. TI Impact Printer/RS-232 Interface optional. Card Plus Disks)
Programming Aids I (TI Impact Printer/RS-232 Interface optional. Disk and Cassette)
Programming Aids II (TI Impact Printer/RS-232 Interface optional. Disk only)
Programming Aids III (TI Extended BASIC cartridge required. Disk only)
UCSD Pascal[10] Compiler (32K Memory Expansion, P-Code Card required. Disk only)
UCSD P-System[10] Assembler/Linker (32K Memory Expansion, P-Code Card required. TI Impact Printer/RS-232 Interface optional. Disk only)
UCSD P-System[10] Editor/Filer/Utilities (32K Memory Expansion, P-Code Card required. TI Impact Printer/RS232 Interface optional. Disk only)
Teach Yourself BASIC[11] (Separate packages for TI-99/4A and TI-99/4. Disk and Cassette)
Teach Yourself Extended BASIC (TI Extended BASIC cartridge required. For TI-99/4. Disk and Cassette)
TI FORTH (Editor/Assembler cartridge and 32K Memory Expansion required. Disk only)
TI Advanced Assembly Debugger (Editor/Assembler cartridge and 32K Memory Expansion required. Disk only)

MATH ENGINEERING

Statistics (Disk and Cassette Data storage system recommended. Cartridge only)
Math Routines Library (Disk and Cassette)
Electrical Engineering Library (Disk and Cassette)
Graphing Package (Disk and Cassette)
Structural Engineering Library (Disk and Cassette)
AC Circuit Analysis Library (TI Impact Printer/RS-232 Interface optional. Disk only)

---

* For TI-99/4A only
1 Developed for Texas Instruments by Microsoft™, Inc. Multiplan is a Trademark of Microsoft, Inc.
2 Developed for Texas Instruments by Aardvark Software, Inc.
3 Developed for Texas Instruments by Pike Creek Computer Company, Inc.
4 Developed by Texas Instruments in conjunction with Scott, Foresman and Company
5 Developed for Texas Instruments by Addison Wesley Publishing Company
6 Developed in conjunction with Scholastic Publishing Company, Inc.
7 Othello is a Trademark of Gabriel Industries, a division of CBS, Inc.
8 A Trademark of Milton Bradley
9 A Trademark of and licensed by Universal City Studios, Inc.
10 UCSD P-Systems is a Trademark of the University of California
11 Developed by Texas Instruments in conjunction with Wolfdata Corporation

---

SOURCES FOR FURTHER INFORMATION

Many different people—men and women with small businesses, teachers, hobbyists, and professionals in many fields—have begun to use home computers. Popular computing magazines now include articles to match their particular interests. Materials about computing are also becoming part of the coverage of many magazines and journals. Teachers' journals now include suggestions for computer use in the classroom, and general magazines also have articles on computing.

Because of the rapid increase in the number of people who see the value and fun in personal computing, no list of suggested sources can cover all information available. Use this list as an introduction to information on computing. A visit to your community's library will show you the wealth of information available on this exciting new field.

Helpful Sources on BASIC

Davis, William S. *BASIC: Getting Started.* Reading, Mass.: Addison-Wesley Publishing Company, 1981.

Dwyer, Thomas A., and Critchfield, Margot. *BASIC and the Personal Computer.* Reading, Mass., Addison-Wesley Publishing Company, 1978.

Inman, Don; Zamora, Ramon; Albrecht, Bob; Quiram, Jacquelyn; O'Dell, Bob. *Beginner's BASIC.* Dallas: Texas Instruments, 1981.

Peckham, Herbert D. *Programming BASIC with the TI Home Computer.* New York: McGraw-Hill Book Company, 1979.

Shelley, John. *Addison-Wesley Pocket Guide to Programming.* Reading, Mass.: Addison-Wesley Publishing Company, 1982.

Texas Instruments. *User's Reference Guide* (for the TI-99/4A Home Computer). Dallas: Texas Instruments, 1981.

General Interest Magazines

BYTE: The Small Systems Journal
POB 590
Martinsville, NJ 08836

Creative Computing
POB 5214
Boulder, CO 80321

99'er Magazine
POB 5537
Eugene, OR 97405

Computers & Electronics
(formerly *Popular Electronics)*
P.O. Box 2774
Boulder, CO

Personal Computing
4 Disk Drive
Box 1408
Riverton, NJ 08077

Popular Computing
POB 307
Martinsville, NJ 08836

Recreational Computing
People's Computer Company
1263 El Camino, POB E
Menlo Park, CA 94025

Journals of Special Interest to Teachers

Classroom Computer News
Subscription Dept.
51 Spring Street
Watertown, MA 02172

The Computing Teacher Journal
The Computer Science Dept.
University of Oregon
Eugene, OR 97403

Computers and Education
Pergamon Press, Inc.
Elmsford, NY 10523

Technological Horizons in Education
(T.H.E. Journal)
Information Synergy, Inc.
POB 992
Acton, MA 01720

Consumer Hotline

For information concerning Texas Instruments Computer Advantage Club classes, for purchasing TI Home Computer software, peripherals, or accessories that you are unable to obtain from your local dealer, or for any questions you may have about your TI products, call Texas Instruments Consumer Hotline at our toll free number **1-800-TI CARES.**

TI-99/4 AND /4A USERS' GROUPS

TI-99/4 and /4A Users' Groups are groups consisting primarily of amateur computer programmers who gather to exchange information, programs and programming ideas.

Below is a list of both national and international Users' Groups locations. If you or a friend are interested in becoming a part of a Users' Group, contact the group in your area by mail, using the addresses supplied.

INTERNATIONAL USERS' GROUPS

International 99/4 Users' Group, Inc.
P.O. Box 67
Bethany, OK 73008

International Home Computer
Users' Association
P.O. Box 371
Rancho Sante Fe, CA 92067

99/4 Users of America
5028 Merit Drive
Flint, MI 48506

AUSTRALIA

National Coordinator:

Shane Anderson
P.O. Box 101
Kings Cross, Sydney N.S.W. 2011

Sydney Interim Coordinator:

Brian Lewis
P.O. Box 149
Pennant Hills
N.S.W. 2110

Melbourne Interim Coordinator:

Doug Thomas
59 Lanstrom Quad
Kilsyth, Vict
Australia 3137

Brisbane Interim Coordinator:

Alwyn Smith
42 Palmtree Avenue
Scarborough, Old
Australia 4020

Perth Interim Coordinator:

Kim Schlunke
P.O. Box 246
Mt. Lawley
Western Australia 6014

Tasmania Interim Coordinator:

Andrew Zagni
161 Carrelast Street
Howrah, Tasmania
Australia 7018

Adelaide Interim Coordinator:

Gerald Tan
Pamela Street
Happy Valley
S.A. 5159

CANADA

Paul Langlois
706-10883 Saskatchewan Drive
Edmonton, Alberta
Canada, T6E 4S6

Carleton Home Computer
Users' Group
John Street R.R.#2
Stittsville, Ontario
Canada KOA 3G0

Toronto Home Computer
Users' Group
3175 Kirwin Avenue
Townhouse #159
Mississauga, Ontario
Canada L5A 3M4

Victoria 99'er Group
402-1471 Fort Street
Victoria B.C.
Canada V8S 1Z4

COLUMBIA

Asociacion Colombiana
de Usarios 99/4
Av. Nutivara #C 3-6
Medellin Colombia S.A.

ENGLAND

T.I. HOME
Paul Michael Dicks
157 Bishopsford Road
Morden
Surry SM4 6BH

GERMANY

Frankfurt

American Express International
Dept. 204
Attn: Mr. C. Quigtar
APO N.Y. 09757

U.S. USERS' GROUPS

ALABAMA

T.I. B.U.G.
709 Nytol Circle
Birmingham, AL 35210

ARIZONA

Arizona 99 Users' Group
4328 E. LaPuenta Avenue
Phoenix, AZ 85044

CALIFORNIA

Orange County 99/4 Users' Group
1673 Chateau
Anaheim, CA 92802

L.A./South Bay 99er Users' Group
5128 Merrill Street
Torrance, CA 90503

SF/South Bay 99er Users' Group
16380 E. LaChiquita
Los Gatos, CA 95030

COLORADO

Colorado 99/4 Users' Group
15177C East Louisiana Drive
Aurora, CO 80012

FLORIDA

Tampa Bay 99er Users' Group
13097 Lois Avenue
Seminole, FL 33542

ILLINOIS

Chicago 99/4 Users' Group
353 Park Drive
Palatine, IL 60067

KANSAS

Mid-America 99/4 Users' Group
P.O. Box 2505
Shawnee Mission, KS 66201

MASSACHUSETTS

M.U.N.C.H
1241 Main Street
Worchester, MA 01603

Pioneer Valley TI-99/4 Users' Group
3 Market Street
Northampton, MA 01060

MINNESOTA

Greater Minneapolis-St. Paul
Home Computer Users' Group
P.O. Box 12351
St. Paul, MN 55112

MISSOURI

Kansas City 99/4A Computer Users
4511 N. Troost
Kansas City, MO 64116

99/4 Users' Group of St. Louis
4127 Quincy
St. Louis, MO 63116

NEW YORK

Jerald Greenberg
34 Maple Ave. Box 8
Armonk, NY 10504

Upstate New York 99/4 Users' Group
7 Steve Lane
Albany, NY 12205

OHIO

Cin-Day Users' Group
11987 Cedarcreek Drive
Cincinnatti, OH 45240

OREGON

Pacific Northwest TI-99/4 Users' Group
P.O. Box 5537
Eugene, OR 97405

Portland Users of Ninety-Nines
421 Northwest 69th Street
Vancouver, WA 98665

PENNSYLVANIA

Daniel Cooper
P.O. Box 285
Hazelton, PA 18201

Pittsburgh Users' Group
P.O. Box 18124
Pittsburgh, PA 15236

RHODE ISLAND

Tri-State Users' Group
P.O. Box 457
Lincoln, RI 02864-0457

SOUTH CAROLINA

Carolina Computer Club
225 Wynchwood Drive
Irmo, SC 29063

TENNESSEE

Athens 99/4 Computer Users' Group
c/o Bob Lamb
McMinn County High School
2215 Congress Parkway
Athens, TN 37303

TEXAS

Dallas TI Home Computer Group
P.O. Box 672
Wylie, TX 75098

Andy Belivacqua
Route 2, Box 75-U
Mansfield, TX 76063

JSC Users' Group (JUG)
15727 El Camino Real
Houston, TX 77062

Houston Users' Group
10107 Westview #112
Houston, TX 77043

Lubbock Computer Club
99/4 Users' Group
5730 67th Street
Lubbock, TX 79424

West Texas 99/4 Users' Group
P.O. Box 6448 M/S 3030
Midland, TX 79701

WASHINGTON D.C.

Washington D.C. 99/4 Users' Group
P.O. Box 267
Leesburgh, VA 22075

WASHINGTON (STATE)

Pudget Sound 99'ers
P.O. Box 6073
Lynnwood, WA 98036

WISCONSIN

Program Innovators
2007 North 71st Street
Waauwatosa, WI 53213

LOGO USERS' GROUP

Young Peoples' LOGO Association
1208 Hillsdale Drive
Richardson, TX 75081

TI BASIC REFERENCE GUIDE

The following information provides a guide to TI BASIC commands (C), statements (S), and functions (F), with examples provided where appropriate.

Some items appearing in this list may not be covered in this manual. For further information, refer to the *User's Reference Guide.*

TERM	EXAMPLE	EXPLANATION
BREAK		Causes program to halt when encountered or optionally when lines listed are encountered. (C,S)
BYE	BYE	Leaves TI BASIC and returns to Master Title Screen. (C)
CALL CHAR	CALL CHAR(36,FFF)	Allows definition of a character to a key. In the example, the computer is instructed to redefine the dollar sign ($) character (ASCII code 36) to a substitute pattern. (C,S)
CALL CLEAR	CALL CLEAR	Clears the screen but doesn't alter the program in computer memory. (C)
CALL COLOR	CALL COLOR(1,16,13)	Allows the change of character color and the corresponding character background block. In the example, the computer is instructed to use white (color code 16) as the foreground color and dark green (color code 13) as the background color for all the characters in character set 1 (refer to Character Set Codes Chart p. 34). (C,S)
CALL HCHAR	CALL HCHAR(1,4,42,10)	Tells the computer to place a horizontal line of characters on the screen. In this case, ten asterisks (character code 42) print on the screen starting at row 1, column 4.(C,S)
CALL KEY	10 CALL KEY(0,KEY, STATUS)	Assigns ASCII code of key pressed on specified key-unit (0-5) to return-variable. (C,S)
CALL SCREEN	CALL SCREEN(9)	Tells the computer to change the screen color, in this case, to medium red (9). (Refer to Color Code Chart p. 34.) (C,S)
CALL SOUND	CALL SOUND (1000,440,1)	Instructs the computer to produce music and noise. In the example, the computer is instructed to play a tone to last one second (1000 milliseconds) at a frequency of 440 cycles per second (middle C on the piano) at a loud volume (refer to Musical Tones Frequencies Chart p. 34). (C,S)
CALL VCHAR	CALL VCHAR (2,5,42,10)	Tells the computer to place a vertical line of characters on the screen. In this case, ten asterisks (character code 42) print on the screen (starting at row 2, column 5). (C,S)
CHR$		Returns the string character corresponding to the ASCII code. (F)
DATA	DATA 34,23,0	Allows data storage within a program. In the example, the numbers following DATA are stored as data to be accessed by a READ statement. (C,S)
DIM	10 DIM X(15)	Dimensions the listed arrays as specified by integers. (C,S)
EDIT	EDIT 20	Displays a line for editing. In the example, line 20 will be displayed. (C)
END	90 END	Stops a program. In a program, END is always the last program statement and must be preceded by a line number, as in the example. (C,S)
FOR-NEXT	10 FOR A=1 TO 10 20 NEXT A	Creates a program loop determined by the equation following FOR. In the example, the value of A is increased by one with each successive loop until the value of A exceeds 10. (S)
FOR . . . TO	10 FOR X=1 TO 100 20 NEXT X	Repeats execution of statements between FOR and NEXT until the control variable exceeds the limit. With each loop between FOR and NEXT, incrementation is by one unless otherwise specified by STEP. (S)
GOSUB	100 GOSUB 500 110 PRINT "OK" 120 END 500 CALL CLEAR 510 RETURN	Transfers control to a subroutine at a specified line number (in this case, line 500) until RETURN is encountered (in this case line 510, which returns the program to line 110). (S)

GOTO	200 GOTO 10	Directs the program to transfer to a line number. In this case, the program will loop back to line 10. (S)
IF-THEN	10 IF A=5 THEN 40	Tests the value of the variable following IF. In the example, if A is equal to 5, the program transfers to line number 40. If A is not equal to 5, the instruction is ignored and the program goes on to the next program line. (S)
INPUT	10 INPUT A	Stops a program and waits for information to be entered. The value entered is assigned to the variable A. (S)
INT	INT(34.6)	Tells the computer to find the greatest integer (whole number) which is less than or equal to the number in parentheses. (F)
LEN	10 LET A$="HELLO THERE" 20 PRINT LEN(A$)	Counts the number of characters in a specific string expression (in this case, 11) and displays the number on the screen. (F)
LET	LET A=5	Assigns a value to a variable. In the example, the value 5 is assigned to the variable A. In TI BASIC, the word LET is optional, so this statement could be written A=5. (C,S)
LIST	LIST	Tells the computer to display the program in its memory. (C)
NEW	NEW	Erases the current program in the computer's memory and prepares to store a new program. (C)
NUM	NUM NUM 5,5	Automatically generates sequenced line numbers beginning at line 100 in increments of 10. Optionally, an initial line and/or increment may be specified. The second example begins numbering with line 5 and increments by five. (C)
OPEN	OPEN #1:"DSK1.MYFILE"	Opens a file, a memory segment on an accessory device, allowing the computer to output data to the device, in this case, Disk Drive 1 (DSK1). (C,S)
PRINT	PRINT "HELLO"	Instructs the computer to print on the screen whatever is enclosed in quotation marks. In the example, the word HELLO is printed on the screen. (C,S)
RANDOMIZE	10 RANDOMIZE	Resets the random number generator to an unpredictable sequence. (C,S)
READ	10 READ X,Y,Z	Tells the computer to find a DATA statement in the program and assign values respectively to the variables which follow the READ statement. (S)
REM	10 REM PAYROLL PROGRAM	Describes or REMarks about program operations at certain points in a program. (C,S)
RESEQUENCE RES	RESEQUENCE RES	Renumbers program statements starting at 100 in increments of 10. Optionally, an initial line number and/or increment may be specified. (C)
RETURN	100 GOSUB 500 110 PRINT "OK" 120 END 500 CALL CLEAR 510 RETURN	Transfers program control from subroutine to statement following corresponding GOSUB (or ON-GOSUB) statement, in this case, line 110. (S)
RND	PRINT RND	Tells the computer to print a pseudo-random number greater than or equal to zero and less than one. (F)
RUN	RUN	Tells the computer to run, or execute, the program in its memory. (C)
SAVE	SAVE CS1 SAVE DSK1.MYFILE	Tells the computer to store the program in its memory onto a cassette tape (SAVE CS1) or diskette (SAVE DSK1.MYFILE). (C)
SEG$	10 LET A$="HELLO THERE" 20 PRINT SEG$(A$,1,5)	Lifts a substring, beginning at a specified position with specified length, from a string expression and prints that segment on the screen, in this case, HELLO. (F)
STOP	500 STOP	Terminates program execution. (C,S)
TAB	TAB(15)	Tells the computer to begin printing at a designated position, in this case, position 15. (F)
TRACE	TRACE	Lists line numbers of statements before they are executed. (C,S)
UNTRACE	UNTRACE	Negates the TRACE feature. (C,S)

GLOSSARY OF PERSONAL COMPUTING TERMS

Allophone—A minimal unit of human speech.

Array—A collection of numeric or string variables arranged in a list or matrix for processing by the computer. Each element in an array is referenced by a subscript describing its position in the list.

ASCII—The American Standard Code for Information Interchange; the code structure used in most personal computers to represent letters, numbers, and special characters.

BASIC—(Beginners All-purpose Symbolic Instruction Code)—A very successful and popular computer language developed at Dartmouth College in 1963-64.

Baud—The signaling speed of information in a computer (typically relating to input and output). It is the number of bits of information per second that a computer can process. Baud rates are a factor in selecting a printer for a computer.

Binary—The two-digit (bit) number system based on 0 and 1. Computers recognize the binary bits 0 and 1 by using gates. Gates are electronic circuits which, when either off or on, represent 0 or 1.

Bit—A binary digit (0 or 1).

Branch—A departure from the sequential performance of program statements. An unconditional branch causes the computer to jump to a specified program line each time the branching statement is encountered. A conditional branch transfers program control based on the result of some arithmetic or logical operation.

Breakpoint—A point in a program specified by the BREAK command where program execution can be suspended. During a breakpoint, operations can be performed in the Immediate Mode (Command Mode) to help locate program errors. Program execution can be resumed with a CONTINUE command, unless editing occurred while the program was stopped.

Buffer—An area of computer memory for temporary storage of an input or output record.

Bug—An error in the hardware or software of a computer.

Byte—A string of eight binary bits. The computer uses approximately one byte of information to encode the letter "A".

Cassette—A form of computer storage for programs and other data.

Central Processing Unit—(CPU)—The nerve center of a computer; the network of electronic circuits that interprets programs and tells a computer how to execute them.

Character—A letter, number, punctuation symbol, or special graphics symbol, usually equivalent to one byte.

Chip—Tiny silicon slices used to produce electronic memories and other circuits. A single chip may have as many as 30,000 electronic parts.

Circuit Board—A rigid fiberglass or phenolic card upon which various electronic parts are mounted. Printer or etched copper tracks connect the various parts to one another.

Command—A word or pair of words that tells the computer to do something in the Immediate Mode. Examples: NEW, LIST, RUN, CALL CLEAR.

Command Cartridge—Pre-programmed ROM cartridges which are easily inserted in the TI computer to extend its capabilities. See also: Solid State Cartridge

Computer—A network of electronic switches and memories that processes data.

Concatenation—Linking two or more strings to make a longer string. The "&" is the concatenation operator.

Console—Main part of the computer containing the keyboard and the CPU.

Constant—A specific numeric real number (such as 1.2 or −9054) or a string constant (any combination of up to 112 characters enclosed in quotes, such as "HELLO THERE" or "275 FIRST STREET").

CPU—See Central Processing Unit.

Cursor—A small flashing square showing where a typed character will appear.

Data—Information, often numerical in form.

Default—A standard characteristic or value which the computer assumes, if certain specifications are omitted within a statement or program.

Disk—See Floppy Disk.

Diskette—See Floppy Disk.

Display—The video screen on the monitor.

Exponent—A number indicating the power to which a number or expression is to be raised, usually written at the right and above the number. For example: $2^8 = 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2$. In TI BASIC, the exponent is entered following the ^ symbol or following the letter "E" in scientific notation. For example: $2^8 = 2\text{^}8$; $1.3 \times 10^{25} = 1.3\text{E}25$.

File—A collection of related data records stored on a device; also used interchangeably with device for input/output equipment which cannot use multiple files, such as a line printer.

Floppy Disk—A flexible plastic disk coated with the same magnetic material used to make recording tape.

Flow Chart—A diagram of geometric shapes connected by arrows that show the progression of a computer program. Flow charts are handy for developing complicated computer programs and illustrating how programs work.

Gate—A very simple electronic circuit that is always either on or off. Clusters of gates can manipulate binary numbers (0 = off, 1 = on). They can also count, do arithmetic, make decisions, and store binary numbers. Gates are the basic building block of computers.

Graphics—Visual construction on the screen, such as graphs, patterns, and drawings, both stationary and animated.

Graphics Line—A 32-character line used by TI BASIC graphics subprograms.

Hard Copy—The permanent print-out of a program or its results produced by a printer connected to a computer.

Hardware—The circuit boards and electronic parts which compose a computer.

Hexadecimal—A base 16 number system using 16 symbols, 0-9 and A-F. It is used as a convenient "shorthand" way to express binary code. For example, 1010 in binary = A in hexadecimal, 11111111 = FF. Hexadecimal is used for constructing graphics characters in the CALL CHAR subprogram.

Immediate Mode—A computer mode in which commands are entered directly into the computer without a line number. Such commands are executed immediately.

Input—The means by which data is entered into a computer—often a keyboard.

Input Line—The amount of data which can be entered at one time. In TI BASIC, this is 112 characters.

Instruction—A statement or command that tells a computer what to do.

Integer—A whole number, either positive, negative, or zero.

Interpreter—The program stored inside a computer that converts or translates BASIC statements into the computer's machine language.

Iteration—One repetition of the technique of repeating a group of program statements. See "Loop."

K—Short for kilo meaning thousand, and used to designate memory capacity—thus a 4K memory has approximately 4,000 storage elements.

Keyboard—A typewriter-like panel of keys used to enter programs and data into a computer.

Line Number—A number identifying a statement in a program. Line numbers determine the order in which a computer executes commands in a program.

Loop—A group of consecutive program lines which are performed repeatedly, usually a specified number of times.

Mantissa—The basic numeric portion of a number expressed in scientific notation. In 3.264E + 4, the mantissa is 3.264.

Memory—Any of the many devices (ROMs, RAMs, floppy disks, magnetic tapes, etc.) that store computer programs and data.

Microcomputer—A computer made by combining a microprocessor with some memory. Microcomputers are small in size, not performance.

Microprocessor—The central processing unit of a computer assembled on a single silicon chip.

Monitor—Television-like device used to display programs as they run or are being written.

Operator—A symbol used in calculations (numeric operators) or in relationship comparisons (related operations). The numeric operators are +, −, *, /, ^. The relational operators are >, <, =, >=, <=, <>.

Output—Information sent from the computer, e.g. graphics on the monitor screen, a report being printed. Also, the means by which data leaves a computer—often a television monitor or printer.

Paper Tape—A narrow ribbon of paper containing computer data in the form of punched holes. A hole indicates the bit 1; no hole indicates the bit 0. Paper tape is sometimes used to enter programs into a computer.

Peripheral—An accessory which can be added to a computer to increase its capability and usefulness (floppy disk, paper tape unit, etc.).

Personal Computer—An economical microcomputer designed for use by small businesses, schools, and computer hobbyists.

Printer—A computer output mechanism that delivers hard copy data.

Print Line—A 28-character line which can be referenced by PRINT and DISPLAY statements.

Program—The list of instructions that tells a computer what to do to perform a task.

Program Line—A line containing a single statement, the maximum length of which is 112 characters.

Programmer—A person who writes computer programs.

Programming Language—Numeric or alphabetic commands which a computer can assimilate, understand, and execute.

Prompt—A symbol (>) which marks the beginning of each command or program line entered; a symbol or phrase that requests input from the user.

RAM (Random Access Memory)—A temporary memory, i.e. one in which data is stored so long as electrical power is applied. Data in RAM can be accessed or changed and is lost if electrical power is cut off.

ROM (Read Only Memory)—Certain instructions for the computer are permanently stored in ROM and can be accessed but cannot be changed. Data in ROM is not lost if electrical power is cut off.

Run Mode—A computer mode in which the computer is executing a program. Run Mode is terminated when program execution ends normally or abnormally. You can cause the computer to leave Run Mode by pressing CLEAR during program execution. (See Breakpoint)

Scientific Notation—A method of expressing very large or very small numbers by using a base number (mantissa) times ten raised to some power (exponent).

Scroll—Movement of text on the screen so that additional information can be displayed.

Software—Computer programs written on paper or stored on magnetic tape or a floppy disk.

Solid State Cartridge—Pre-programmed ROM cartridges which are easily inserted in the TI computer to extend its capabilities. See also: Command Cartridge

Speech Synthesizer—A peripheral that enables the computer to talk.

Sprite—A character to which you can give shape, color, speed, screen position, and direction in TI LOGO and in TI Extended BASIC.

Statement—A single line of a computer program containing a single instruction such as PRINT, LET, RUN, etc.

String—A series of letters, numbers, and symbols treated as a unit.

Subprogram—A predefined general-purpose procedure accessible to the user through the CALL statement in TI BASIC. Subprograms extend the capability of BASIC and cannot be easily programmed in BASIC.

Subroutine—A program segment which can be used more than once during the execution of a program, such as a complex set of calculations or a print routine. In TI BASIC, a subroutine is entered by a GOSUB statement and ends with a RETURN statement.

Subscript—A numeric expression which specifies a particular item in an array. In TI BASIC the subscript is written in parentheses immediately following the array name.

Terminal—An input device such as a keyboard, or an output device such as a printer or a TV monitor.

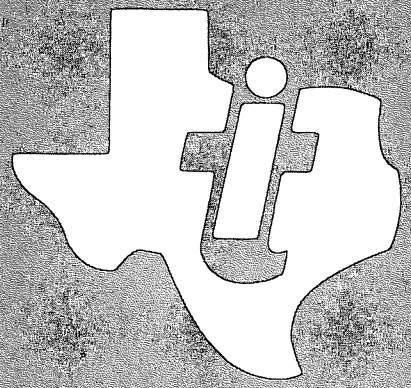
Trace—Listing the order in which the computer performs program statements. Tracing the line numbers can help you find errors in a program flow.

Turtle—In TI LOGO, the small triangle with which designs are drawn on the screen.

Users' Group—An informal or formal association of persons who own or operate similar or identical computing equipment. Users' groups are usually formed to exchange programs and other helpful information.

Variable—A name given to a value which may vary during program execution. A variable is a memory location where values can be replaced by new values during program execution.

Wired Remote Controllers—Small, handheld controls, sometimes called joysticks, used to move graphics in desired directions on the screen.



TEXAS INSTRUMENTS

This certifies that

_____

has successfully completed the course of instruction in

BASIC PROGRAMMING FOR ADULTS

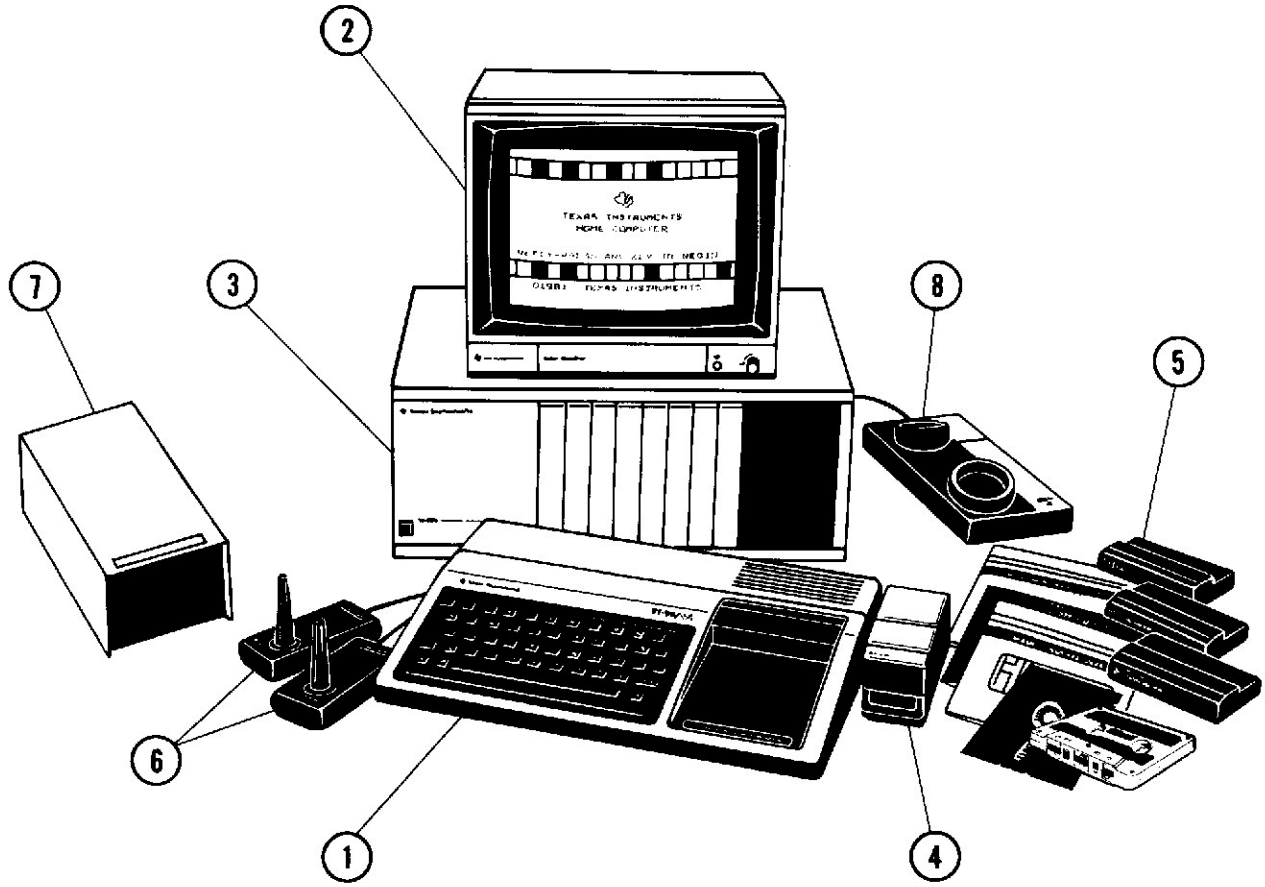
sponsored by the

TEXAS INSTRUMENTS COMPUTER ADVANTAGE CLUB

_____ _____

Program Instructor Date

COMPUTER COMPONENTS



1. **TI-99/4A HOME COMPUTER CONSOLE:** A typewriter-like console that allows you to enter, store, and manipulate data.

2. **VIDEO MONITOR:** A ten-inch color screen with a display format for 24 lines of 32 characters and audio capabilities.

3. **PERIPHERAL EXPANSION SYSTEM:** A compact system designed to centralize the Disk Memory System, the RS-232 Interface, the Memory Expansion unit, and other accessories.

4. **SPEECH SYNTHESIZER:** A device which reproduces human speech electronically and accurately, allowing the computer to communicate verbally.

5. **HOME COMPUTER SOFTWARE:** A large library of pre-programmed cassettes, diskettes, and Solid State Cartridges designed to help you learn, keep household records, or play stimulating games.

6. **WIRED REMOTE CONTROLLERS:** Eight-position remote controls with top-mounted action button which allow you to move objects on the screen.

7. **DISK MEMORY SYSTEM:** Stores data or programs for later use.

8. **TI TELEPHONE COUPLER (MODEM):** Allows the Home Computer to send or receive information through a telephone.

TEXAS INSTRUMENTS

Printed in U.S.A.

1053666-0001